3-21-2013

Modeling Cyber Situational Awareness through Data Fusion

Evan L. Raulerson

MODELING CYBER SITUATIONAL AWARENESS THROUGH DATA FUSION

THESIS

Evan L. Raulerson, Captain, USAF

AFIT-ENG-13-M-41

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-13-M-41

MODELING CYBER SITUATIONAL AWARENESS THROUGH DATA FUSION

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Evan L. Raulerson, B.S.C.S.

Captain, USAF

March 2013

AFIT-ENG-13-M-41

MODELING CYBER SITUATIONAL AWARENESS THROUGH DATA FUSION

Evan L. Raulerson, B.S.C.S.
Captain, USAF

Approved:

Kenneth Hopkinson, PhD (Chairman)	Date
Maj Kennard Laviers, PhD (Committee Member)	Date
Timothy Lacey, PhD (Committee Member)	Date

AFIT-ENG-13-M-41

Abstract

Cyber attacks are compromising networks faster than administrators can respond. Network defenders are unable to become oriented with these attacks, determine the potential impacts, and assess the damages in a timely manner. Since the observations of network sensors are normally disjointed, analysis of the data is overwhelming and time is not spent efficiently. This time would be better used to determine and implement the best responses to network events to minimize damage. Automation in defending cyber networks requires a level of reasoning for adequate response. For the automated response systems that exist, they are mostly limited to scripted responses based on data from a single sensor. Better defense tools are required. For network administrators and automated decision-making agents to develop suitable response plans, they must know the correct and current information about the network. This research develops a system framework that aggregates data from heterogeneous network sensors, including intrusion detection systems, antivirus software, network mapping tools, and host monitoring software. The collected data is correlated into a single model that is easily interpreted by decision-making entities. This research proposes and tests an impact rating system that estimates the feasibility of an attack and its potential level of impact against the targeted network host as well the other hosts that reside on the network. The impact assessments would allow decision makers to prioritize attacks in real-time and attempt to mitigate the attacks in order of their estimated impact to the network. Additionally, this system provides the range of potential actions that an administrator would be allowed to perform on the network. For automated agents, the inclusion of the action set reduces the search space, allowing quicker results. The ultimate goal of this system is to provide computer network defense tools the situational awareness required to make the right decisions to mitigate cyber attacks in real-time.

To my adoring wife and children. Thank you for supporting me every step of the way and for the sacrifices you made during this research.

Acknowledgments

I would like to express my sincere appreciation to my wife. She always stood by me and gave me encouragement to overcome the stressful times. I am eternally grateful for her patience and inspiration.

I am thankful to Captain James Hannan and Captain James Emge for helping test the output of my system. Since their research systems used the data produced from the tools that I built, they further drove my system's requirements. Additionally, Captain Emge helped set up the test environment and the attack scenarios for the experiment. The feedback and assistance they gave me helped make my system more usable and easily interoperable with other systems.

Finally, I would like to thank my advisor, Dr. Kenneth Hopkinson, and my committee, Dr. Timothy Lacey and Major Kennard Laviers. Their guidance and mentorship were invaluable during this research.

Evan L. Raulerson

Table of Contents

Page

Page

Page

List of Figures

Figure Page

List of Tables

List of Acronyms

Acronym	Definition
AFIT	Air Force Institute of Technology 103
AI	artificial intelligent 64
AOI	Activities of Interest 77
AFRL	Air Force Research Laboratories 102
ARMOUR TDP	Automated Network Defense Tech Demonstration Project 11
ARP	Address Resolution Protocol 9
COTS	Commercial-Off-The-Shelf 31
CPU	Central Processing Unit 68
CVE	Common Vulnerabilities and Exposures 106
CVSS	Common Vulnerability Scoring System 45
DIR	Data-to-Information Ratio 85
DMZ	DeMilitarized Zone 69
DNS	Domain Name Service 68
DoD	Department of Defense 107
DoS	Denial-of-Service 83
DDoS	Distributed Denial-of-Service 83
FTP	File Transfer Protocol 83
GB	Gigabyte 68
GHz	Gigahertz 68
GIG	Global Information Grid 1
HTTPS	Hypertext Transfer Protocol–Secure 9
ICMP	Internet Control Message Protocol 8
IDS	Intrusion Detection System 103

Acronym	Definition
IP	Internet Protocol 79
IPSec	Internet Protocol Security 9
IT	information technology 103
JDL	Joint Director's of Laboratories 105
LAN	Local Area Network 107
MB	Megabyte
MDP	Markov Decision Process 17
NAIR	Novel Automated Intrusion Response 17
NIPRNet	Non-Secure Internet Protocol Router Network 2
NVD	National Vulnerability Database 54
OODA	Observe, Orient, Decide, and Act 107
OpenVAS	Open Vulnerability Assessment System 105
OPSEC	Operations Security 8
OS	Operating System 104
PHP	(Personal home page) Hypertext Preprocessor
ROC	Receiver Operator Characteristic 94
SIPRNet	Secret Internet Protocol Router Network 2
SMB	Server Message Block 83
SQL	Structured Query Language 84
SSH	Secure Socket Host 35
SUT	System Under Test 83
S/MIME	Secure/Multipurpose Internet Mail Extensions 9
TB	Terabyte 68
TVA	Topological Vulnerability Analysis 19
UPnP	Universal Plug and Play 54

Acronym	Definition
VM	Virtual Machine 68
XML	eXtensible Markup Language 31

MODELING CYBER SITUATIONAL AWARENESS THROUGH DATA FUSION

I. Introduction

1.1 Overview

The Department of Defense (DoD) and industry partners have placed many investments in building their cyber infrastructures, but as the footprint grows, so does the number of attack vectors for malicious actors. Cyberspace is plagued by a perpetual barrage of cyber attacks from countless directions; this makes cyberspace a difficult domain to defend. The threat of cyber attacks or malware infections are a constant threat from government and commercial businesses alike. Per the Symantec Intelligence Quarterly: July - September 2011 [51], Symantec observed from their sensors 155 million cyber attacks during the quarter. In addition, Symantec estimates that between 80,000 and 100,000 web-based attacks occur every day. Many of these attacks go unnoticed by their victims. The Symantec 2012 summer report [53] details that about 45% of all cyber attacks are targeted at the defense industry; even small businesses are affected: they are struck with about 36% of all cyber attacks. This growing problem must be reduced.

In 2009, then Secretary of Defense Robert M. Gates [13] testified to the Senate's Armed Services Committee that the Armed Forces' Global Information Grid (GIG), the United States military's combined computer networks, are under threat of attack. He added that it takes as little as "cheap technology and minimal investment" to become a serious threat in cyberspace. Former SecDef Gates estimated that the GIG contained more than 15 thousand networks and close to 7 million computers. Four years later, the present size of the GIG has undoubtedly increased. Against a network as large as the GIG, many attacks go unobserved.

The uncertainty of cyberspace activities has only become more clouded over time. In 2010, William J. Lynn, the former Deputy Secretary of Defense, described in the Pentagon's Cyberstrategy the events that the led to Operation Buckshot Yankee. He told of a rogue thumb drive that was plugged into a U.S Central Command computer system. From the thumb drive, a virus spread through the Non-Secure Internet Protocol Router Network (NIPRNet) and the Secret Internet Protocol Router Network (SIPRNet) and opened backdoors for unauthorized agents to steal sensitive information. If the network administrators had more oversight of what was occurring within their networks, the damage would have been greatly reduced.

To better defend in cyberspace, cyber defenders must first have situational awareness of their respective cyber networks. Dr. Eugene Schultz explained in an article on the Network Security Consulting Blog [44] that network situational awareness is underrated and essential to accurately defend in cyberspace. A key point he made was that administrators are bombarded with too much information to derive the necessary details to defend a network. He focused on the events of the TJX Company network breaches, which occurred for eighteen months [44]. Dr. Shultz argued that with proper situational awareness, the holes in the network could have been closed and the intruders stopped before they acquired over 45 million credit card numbers. Monetary gains may appear to be the driving force in attacks on civilian networks, but there are other driving forces working against government and military networks.

The Air Force has already acknowledged that cyber situational awareness is a vital component of cyber warfare and is direly needed. Dr. Mark Maybury describes in the Air Forces Cyber Vision for 2025 [26] that activities in cyber war will definitely escalate. In the prescribed roadmap, Dr. Maybury emphasized the need for situational awareness and battle damage assessments for the domain. The goal is to gain a foothold in establishing

situational awareness and data fusion for the cyber domain between the years 2012 and 2020.

1.2 Problem Statement

As stated, a major shortfall in defending cyberspace is that the domain is difficult to analyze; there is a lack of situational awareness in computer networks. The primary cause of this is that sensors are disjoint. Related data amongst different objects are not linked, making the big picture difficult to see. This forces administrators to cross reference the information, costing excessive time that can be efficiently used elsewhere.

Additionally, most research and development in cyber warfare uses simulated network environments as the underlying system. These simulated networks comprise of scripted data that resemble real networks in some way. Results from tests that were run on such systems are not given the same recognition as tests from actual networks since simulations cannot account for all of the variances that a real network creates. In addition, systems developed for a simulated network often encounters problems when converting to a real network. These systems must often be redesigned when they do not interface correctly to actual networking systems.

Furthermore, tools for network management, such as visualization and decision-making tools, require network situational awareness that includes prioritized events and impact assessments for the known network assets [47]. For research and development of systems that depend on situational awareness, there are no off-the-shelf products or simple approach that can be used. Developers must start from scratch and develop a cyber situational awareness system to support their tools or research. Most often the time spent building this system is too overwhelming for many, thus simulated systems are used instead. A prebuilt system or a simple approach with guidelines and setup instructions are essential for the research and development community to advance cyber situational awareness and defense.

1.3 Goals and Approach

This research has several goals that will be pursued in order to reach solutions to the aforementioned problems.

First, a system architecture and framework will be designed and implemented that provides situational awareness of a computer network. The design will be supported and adhere to past and current research in the cyber situational awareness domain, primarily stemming from the Joint Director's of Laboratories (JDL) data fusion model [64] and research by the Air Force Research Laboratories (AFRL). This system will aim to be a simple framework for achieving situational awareness in a real Local Area Network (LAN).

From the designed framework, this research will result in a tool for aggregating data from heterogeneous sensors to create a real-time network model. The model will detail network traffic events, network asset discovery and inventory, network host monitoring, and threat impact assessments.

To assess the validity of the system, it will be installed in a real network and tested with actual cyber attacks. The environment will consist of a virtual network that contains various client and server Virtual Machine (VM)s. The modeling software will collect data from the network and provide actual on-going assessments of the network.

1.4 Contribution

This research will contribute to the United States Air Force and to the information technology (IT) community in several ways. First, a process will be developed to guide researchers to achieve basic cyber situational awareness of LANs they possess. This process will be simple to follow as it is aimed to reach a situational awareness state quickly so that researchers can focus on building upon the situational awareness framework or other cyber defense research.

In addition to the process, a tool that adheres to the process will be developed. Air Force researchers will have this tool available, enabling them to accelerate their research.

Furthermore, the process and the tool will contain methods to determine in real-time the impact of attacks in regards to actual computers in the network. An impact scoring system will be proposed and tested that ranks the severity of attacks. The impact assessment process will reduce the time in responding to attacks by showing administrators what they should be concerned with and what information is be irrelevant.

Ultimately, this research will create a foundation for future research to be able to use a real network and tools to pursue development and advancements in cyber attack and defense.

1.5 Summary

This chapter explained that a greater network situational awareness is needed to accurately defend in cyberspace in a timely manner. A network modeling system will be developed and tested to solve this problem.

This tool will provide the right information to network administrators, visualization tools, or automated agents to protect and defend their networks.

To progress in the research and solve the problem statement, four phases must be addressed and completed. These phases are detailed in chapters two through five. Chapter two explains the current and past research in defending cyberspace, achieving network situational awareness, methods of data fusion, and modeling the situational awareness for further use and analysis. Chapter three contains the design of the system and how it was developed. Chapter four outlines how the system will be tested and how the metrics will be achieved. In chapter five, the results of the experiment is revealed and statistics will be applied to show how accurate the network modeling system is in identifying cyber attacks. In addition to these four chapters, chapter six will conclude and summarize the research and experiment.

II. Literature Review

Researchers have been scrutinizing the cyber situational awareness problem since the late 1980s [64]. As new requirements arise for defending cyberspace, the lack of proper cyber situation awareness becomes more apparent. Presently, information technology (IT) networks require a cognitive reasoning and planning system that is able to implement real-time defensive measures. Tools that emphasis network visualization and automated response depend on refined cyber situational awareness inputs [47]. Shiravi et al. [47] explain that situations and critical network events must be prioritized for such administration tools. In order for systems to determine the best response to a cyber attack, respond to the attack, and learn from its successes and mistakes, a cyber situational awareness framework must be in place that provides the necessary data at the right time [47]. This chapter will focus on describing tools, processes, and research in cyber situational awareness and data fusion.

In examining automated cyber defense, specific categories of research must be analyzed. The field of cyber warfare must be assessed to determine cyber attack vectors and appropriate defensive responses. This chapter will emphasize competing data fusion techniques and avenues. In addition, this chapter will explore the requirements for making decisions and predicting future states of a network in support of cyberspace defense; this includes the areas of artificial intelligence that encompass state modeling and heuristics.

2.1 Cyber Warfare

Cyber warfare covers topics such as the actions of network attack and network defense. These attack and defense actions compete against one another in computer networks. To effectively win requires that one understand their own and their opponents' decision making

processes [7]. This section analyzes cyber warfare techniques and fundamental situational awareness.

2.1.1 Cyber Attack Vectors.

In cyber warfare, the actions of black hat hackers are a major concern of computer network administrators [9]. A "black hat" type of hacker is a network user that exploits network vulnerabilities for personal gain or for the sake hurting others [48]. To adequately defend against malicious attacks, defensive systems must have situational awareness of the domain [47]. The following sources identify attack vectors that must be considered to defend a network appropriately.

Black hat agents generally follow some sort of methodology when trying to exploit or compromise a computer system, as identified by Dr. Eric Cole [9]. He describes the attack methodology of attackers as being broken into specific phases: scanning, exploitation, maintaining access, and covering tracks. In scanning, networks are probed for vulnerabilities. This is the phase in which the attacker maps the network and determines which ports are open on the machines. In the exploitation phase, attackers attempt to gain unauthorized access to computer systems and network traffic. This includes pushing malicious code to the machines, traffic spoofing, man-in-the-middle attacks, replay attacks, system scanning, and software exploitation. The next phase, maintaining access, involves the attacker creating a presence on the host machines and developing a backdoor for quick access. In covering tracks, the attacker hides their presence on the host, which may include rootkits.

Ed Skoudis and Tom Liston explain other attacks in their book [48] such as password cracking and rootkits. Password cracking occurs in the exploitation phase, while rootkits help an attacker maintain their presence on a system and cover their tracks. Furthermore, Zheng Wu et al. [65] identify four primary means of attack: information gathering,

privilege escalation, illegal file modifications, and resource exhaustion. From these sources, the attack vectors can be categorized into steps as seen in the following table:

Table 2.1: Cyber attack methods

Category	Attack
Scanning	Network mapping Port scanning Vulnerability scanning
Gaining access/exploitation	Spoofing Password cracking Software exploitation Structured Query Language (SQL) injections
Maintaining access	Rootkits Backdoors
Covering tracks	Rootkits Illegal file modification

In the cyber attack methodology, reconnaissance cannot be observed by network sensors; this crucial step is normally fulfilled outside of the target network [48]. Mitigating the leverage that black hat agents gain from reconnaissance can be minimized with Operations Security (OPSEC), the practice of minimizing accidental disclosure of critical information. With the exception of the reconnaissance phase, the other phases connect the attacker to the network and can be observed by network sensors. The attacks described in the aforementioned phases should be considered when protecting computer networks.

2.1.2 Cyber Defensive Maneuvers.

To defend against cyber attacks, the network administrators must establish defensive measures and prepare courses of action to use against such attacks. In their book, Skoudis and Liston detail methods [48] to address or respond to cyber attacks. The following list summarizes Skoudis and Liston's methods for securing IT networks against these attacks.

Network mapping To suppress the attackers' ability to learn the configuration of the local network, the network gateways should block Internet Control Message

Protocol (ICMP) messages and disable ”Time Exceeded” messages. This limits the information gathering ability of attackers; conversely, it can make managing a network more difficult for administrators.

Port scanning Unused ports and their services should be disabled to reduce the information gained during port scans. All open ports are potential egress points for attackers. Minimizing the number of open ports limits the attack vectors into and through a network.

Vulnerability scanning Administrators should remove vulnerabilities from their systems whenever possible. Systems should be patched and unused ports closed. System administrators should use the same type of tools as the attackers would use to scan their own machines; periodic scans can reveal new vulnerabilities to administrators over time.

Password cracking Automated tools exist that can brute-force or crack passwords for network accounts and encrypted files. Administrators should institute strong password policies across their networks that aim to make using automated cracking futile. These policies should include minimum character length and expiration dates for passwords.

Sniffing attacks Network traffic can be encrypted to prevent outsiders from reading and exploiting network traffic. Using Internet Protocol Security (IPSec), Hypertext Transfer Protocol–Secure (HTTPS), and Secure/Multipurpose Internet Mail Extensions (S/MIME) can greatly harden the network. Switches should be used instead of hubs, since hubs broadcast everyone's' traffic to all connected computers, while switches only send traffic to the intended destination. The use of static Address Resolution Protocol (ARP) tables on host machines can help prevent ARP cache poisoning, which is a technique for man-in-the-middle attacks.

Internet Protocol (IP) spoofing IP spoofing is a technique used to hide one's IP address when performing unauthorized actions on a network. To minimize IP spoofing, firewalls should have inbound and outbound filters to prevent anomalous traffic. This traffic may include packets coming or leaving the network with an incorrect source IP address; packets leaving the network should have a source IP address of that network interface, while packets entering the network should not have a source IP of that network. Systems should be patched so that the latest techniques are employed to prevent predictions of the packet sequence numbers, which allow attackers to insert illegitimate packets into network traffic.

SQL injections Black hat users can take advantage of exploits in public-facing web servers that connect to internal databases. By altering web page scripts or inserting unintended scripts into web page forms, the server may disclose critical information from the database to the attacker. These SQL injection attacks towards web servers can be prevented or minimized by having the web application filter user input, limit the web application's permissions, and utilize parameterized queries.

Malicious software Backdoors, spyware, and viruses can be mitigated by running antivirus applications on the hosts, which can quarantine infections when found. Rootkits can be discovered by fingerprinting the software in the system and periodically validating the integrity of the software. Removing the infected program may not completely remove the threat. The best way to recover from a rootkit is to reinstall the system from scratch or restore the system to a clean backup.

To add to Skoudis and Liston's recommended response techniques, Wu et al. [65] add additional methods. In their network defense model, Wu et al. describe basic responses to intrusions in their system to include powering off the host, rebooting the host, killing a connection, restarting a connection, killing a process, restarting a process, modifying

privileges, modifying file attributes, disabling a user, or no response. These actions should be considered when identifying the full range of possible responses to cyber attacks.

2.1.3 Warfare decision making.

To adequately defend against cyber attacks, administrators must know when and how attacks are penetrating their defenses [47]. This knowledge of what is happening in their network or environment is situational awareness [55]. In cyber warfare, computer networks are treated as the battlespace. Similar to land, air, and sea battlespaces [37], the factors of the cyber battlespace must be identified and presented to the cyber warfighter.

Most research in cyber situational awareness stem from military situational awareness doctrine. For instance, the reference [59] identifies that a network that can adapt itself should be defined as a network that "has a cognitive process that can perceive current network conditions, and then plan, decide, orient, and act on those conditions." This decision making process stems from the Observe, Orient, Decide, and Act (OODA) loop developed by John Boyd [7]. This feedback loop from airpower doctrine describes four phases to formulating a best response to a situation–observe, orient, decide, and act. This process has been widely embraced and adapted to various domains, including information networks.

Building upon the OODA loop, Sawilla and Wiemer [43] propose a process called Automated Network Defense Tech Demonstration Project (ARMOUR TDP) that aims to automate network defense. This situational awareness process was developed to be used in military computer systems. In addition, Klein et al. [20] described their own approach to fit cyber defense into the OODA loop. Combined, these two approaches adapted cyber into the OODA loop per the following guidelines:

- The "observe" phase is satisfied by using sensors across the network. Tools such as host monitoring software, intrusion detection systems, anti-virus scanners, and firewall logging can collect data about a network.

- In the ”orient” phase, the information from the sensors is prioritized and associated with data and policies already known in the system. Data can be organized into a classification scheme and combined to form a better picture of the environment.

- In the ”decide” phase, the system chooses an action that is most appropriate in resolving a problem or optimizing for the network. The decision should take into account the data collected and the estimated intent of the attacker. This action should be optimal for the network and least optimal for attackers.

- The ”act” phase simply involves fulfilling the newly decided change to the network. The system implements the chosen action to the applicable areas of the network.

This section defined and described leading approaches in cyber situational awareness. In developing a system that would provide the current state of a network, network data must be collected or observed and then oriented in a way that is simple to read and understandable.

2.2 Gathering network information

The first step in developing cyber situational awareness, as stated in the previous section, is to gather data from the network. There are many tools in the commercial market and services provided by government organizations that provide state-of-the-art information gathering and analysis techniques. The following sensors and tracking systems set the foundation for a cyber defense posture, but combined, they can present information overload [4]. In addition, not all of the data necessary to make the right decisions are provided by a single sensor, so sensor data must be aggregated [4]. The tools in the following subsections are vetted by the information technology security community [31] [45] as the best in their categories.

2.2.1 Detection mechanisms.

Companies sometimes shy away from implementing intrusion detection systems on their networks due to the perception that they are expensive and difficult to deploy [8]. Several Commercial-Off-The-Shelf (COTS) products make this perception false; these products are flexible enough to be installed in various network topologies, simple to use, and are either free or low-cost. These robust products have already been developed to identify intrusions, anomalies, and vulnerabilities in computer networks. The following popular detection systems can be easily deployed in networks and are able to interface with new systems.

Snort® Developed by Sourcefire [49], Snort is an Intrusion Detection System (IDS) that sniffs network traffic. It is able to indentify many cyber attacks attempsts to include scanning and exploitation.

Nmap Created by Fyodor Vaskovich (Gordon Lyon) [33], this is a network exploration tool that identifies details on hosts. It can fingerprint operating system versions, running services, available ports, and the filters that are on the host. During scans, the host running Nmap will send packets to a target machine. Depending on the type of scan Nmap is performing, thousands of packets may be sent across the network, which is very noisy and may alert IDSs.

Nessus Nessus is a powerful host vulnerability scanner, developed by Tenable Network Security® [57]. It identifies open services on a machine, the services' vulnerabilities, and the exploits used to take advantage of the vulnerabilities. Like Nmap, it probes a target machine with numerous packets that can be noisy. Nessus references its online database, which is updated daily for the latest attack vectors.

Nagios® This is a monitoring system by Nagios Enterprises that observes network health, to include network protocols, host services, server health, and applications [30]. The

Nagios Core is a system to be installed on a server, which receives inputs from deployed Nagios services on host systems. The Nagios Core builds a picture of the hosts' statuses and provides administrators alerts to activities in their networks. This system is open source and is extensible to third-party programs.

Open Vulnerability Assessment System (OpenVAS) OpenVAS is a set of tools created to scan networks for vulnerabilities and manage the results [35]. OpenVAS is completely freeware. It is built to be used in Linux, but also has limited compatibility with Windows. In addition, the OpenVAS framework and system updates the scanner with new tests daily and provides a community bug tracking service.

Nexpose® Nexpose is a vulnerability scanner developed by Rapid7, the creators of Metasploit® [38]. Nexpose has a community version that is free, which was designed for networks having less than thirty users. Also, they have an enterprise version for larger networks. Rapid7 describes the community version as being able to scan networks, databases, and operating systems. The enterprise version has the same features, but it can also scan virtualized machines and create custom reports. Both can operate on Linux or Windows.

SAINT® Developed by SAINT Corporation [40], SAINT is a popular vulnerability scanner and penetration tool. SAINT Corporation touts that it can assess a plethora of devices and is compliant across government and Department of Defense (DoD) standards. A paid subscription is necessary to use the SAINT cyber security tools.

These aforementioned tools are vetted by the computer security industry and are considered best and most reliable [31] [45]. Since the purpose and output of these tools give administrators different views of the network, a compilation of selected tools are needed to build an operational picture. Correlation of data from various heterogeneous tools such as these would give network administrators the necessary data to defend their networks [34].

The choice of sensors is only one factor in data acquisition; sensor placement is another factor. There are different approaches to how the sensors and monitors should be deployed across a network. Technical report 2010-078 by Air Force Research Laboratories (AFRL) [17] discuss their choice of optimal placement of sensors on a network. In their assessment, it was found to be better to focus placing sensors over mission critical assets versus wasting resources on analyzing every machine. They focused on critical paths and considered alerts on those paths to be higher than other alerts.

In contrast to AFRL's approach, the paper by Chengchen et. al [15] described a strategic sensor placement across a network. They emphasized that sensors be equally distributed across the network versus at the entry points. They simulated a ten-node network and found the distributed approach to be better than having sensors solely at the gateway. Since the test by Chengchen et. al was not compared against an asset-based approach, as proposed by AFRL, a superior approach cannot be derived. AFRL's approach was focused on cost and resources, while the research by Chengchen et. al focused on detection rates, so the research may not be accurately compared.

2.2.2 Vulnerability databases.

Network sensors are not the only data sources needed to explore situational awareness on a network. Network data must be analyzed for vulnerabilities, which requires data on vulnerability requirements. Just as networks constantly change, attack vectors change as well. As new attacks are discovered each day, network analysis tools must remain up-to-date on the latest vulnerabilities and exploits [9]. There are several public databases that track vulnerabilities and detail patching measures for known exploits. The following list describes popular exploit and vulnerability databases:

Common Vulnerabilities and Exposures Common Vulnerabilities and Exposures (CVE)®[29] is an online system of tracking computer vulnerabilities and patch management. CVE

imposes identifiers on vulnerabilities, which are widely used by other systems in tracking bugs.

National Vulnerability Database (NVD) Hosted by the National Institute of Standards and Technology, the NVD [32] provides an online repository of vulnerabilities. NVD supports the CVE identification system.

Bugtraq Bugtraq is an online database of vulnerabilities, which is hosted by SecurityFocus™ [46]. It strives to bring the IT community together to discuss and share awareness of computer security vulnerabilities.

DeepSight™ Threat Management System DeepSight is a security tool by Symantec™ [52]. It provides an online dashboard of cyber threats across the globe, from data discovered by the DeepSight Extractor tool used by its customer base.

The information from these databases can provide a network situation awareness system the necessary vulnerability assessment data to expose many of the holes in modern networks. In addition, as new information is discovered, network tools should reference these sources often to ensure a defensive system is up-to-date and does not error in identifying known vulnerabilities. COTS IDSs must also be able to automatically update with the information these sources provide.

2.3 Cyber attack vectoring prediction

In depth research has been conducted to understand and map cyber attack vectors and methodologies. Identifying an attack that is occurring is only the first step. Further analysis must be accomplished to determine what else is left for the attacker to pursue and what the goal might be. It may be difficult to determine the precise intentions of an attacker, but there are trends that show the probabilities of subsequent actions. The following subsections describe techniques in portraying a network and determining potential subsequent states.

2.3.1 State Modeling.

In order for the network to be evaluated and its future states predicted, it must be modeled as a snapshot in time. Models must be designed to have only the information needed by the system. Too many features will cost unnecessary resources when performing searches, while too few features will portray an incomplete model. The following sources explain effective ways to portray a network with minimal features.

Milner [28] describes the concept of a derivation tree, which is a collection of an expression's derivatives, where $E \rightarrow \alpha \rightarrow E'$. In this, E represents the expressions of an agent's behavior, while α represents the transition of an action. These derivation trees appear related to Markov Decision Process (MDP). Milner further expressed the states of agent configurations as algorithms. The foundation of the work built upon the simple concept of a transfer of information as between a sender and receiver through a medium.

Described by Russell and Norvig [39], a simpler method of expressing the transition between states is to look at the state transition diagram. This shows that actions occur as transitions between state models, seen in the diagram below:

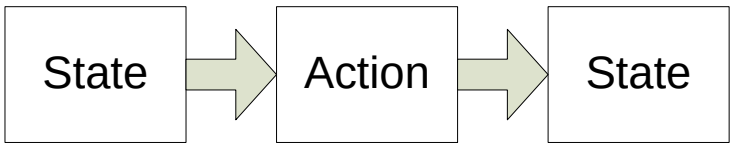

Figure 2.1: State Transition Diagram

Russell and Norvig [39] further explain MDP as a process that can decide on an optimal state based on a previous state in a stochastic environment. The use of MDPs in cyber warfare research has been promising. The Novel Automated Intrusion Response

(NAIR) system, by Xin Zan et al. [68], is modeled by MDPs. In the system, actions are associated with probabilities. NAIR has three modules; one module generates states, another predicts attacker intent from the states, and the final module creates a policy or plan of action based on input from the other two modules. The system determines the probability of states using forward chaining in the MDPs. Furthermore, Fava et al. [11] provided an approach to characterize and predict cyber attacks through MDPs. The approach aimed to predict events in multi-stage attacks by identifying actions that correspond to steps in an attack graph. From these resources, MDPs provide a structured means to predict and identify actions on a computer network.

Reference [59] explains that modeling networks into states can be complex and resource intensive. Though, all of the system's details are unnecessary for a network manager to make a decision to protect or optimize the network; a smaller model of the network is sufficient. Lye and Wing [23] describe their methods in portraying a computer network. Their features include the network hardware, software on the machines, how the computers are connected to one another, and privileges of users on the devices.

Upon gathering and correlating the necessary data to accurately model a network, it has to be presented in a way that is flexible enough for different automated systems to analyze and update. Fava et al. [11] describe cyber attacks in eXtensible Markup Language (XML) format. With XML, they wrapped the data in tags. They followed an attack by listing all of the indicators within a set of "Track" tags. The child tags included alert, sensor, category, and description data. Upon completely applying network data to the tags, the contained data formed an extensive picture of the network. XML format is a robust method of organizing data and easily sharable between different platforms [63]. This method shows the feasibility to model a complete network state in XML.

2.3.2 Attack Trees and Graphs.

Attack trees, described by Tran and Jin [61], are classification models that identify objects by their characteristics. Each node of the tree contains a different conditional rule that moves the object to one of its leaf nodes. Upon being mapped by the tree, the object would contain a new classifier.

Attack trees and graphs are useful to indentify attacks on a network [60]. Once an ongoing attack is identified, predictions can be made as to which actions the attacker may choose next. Tidwell et al. [60] propose an extended attack tree that identifies the dependencies of actions for a cyber attack to occur. This method details the steps of an attacker to achieve a desired goal. The attacker's potential goals would reside in leaf nodes; the actions of a current state would map to a node on the tree and would show the possible outcomes in its child nodes. Tidwell et al. further show that it is beneficial to create dependencies on parent and child nodes and add parameters to the nodes that aided in determining the most likely path that an attacker would take.

AFRL's technical report [17] describes their attack graph approach called Topological Vulnerability Analysis (TVA). This method maps the dependencies of vulnerabilities and shows the paths that attackers would use to exploit a network. This method provides the defensive strategies that should be used in mitigating the attacks. One such strategy described is the optimal placement of sensors on the network. Based on what the attack graph knows, it can reduce false positive alerts from the IDS. The goal of this research is to fortify a network prior to attacks.

The concepts of attack trees and graphs are similar to dependency tables. Tree and graphs can be viewed as a graphical representation of rows that point to other rows in a table. This concept can be expanded to be useful for other network strategies. It would be just as simple to create a tree that supports defensive postures and network health as it is to map network attacks.

2.4 Data Fusion

To capture the entire picture of network health and cyber attacks, data will have to be retrieved from distributed sensors, likely of different types. The acquired information must then be merged into a single model; this merging of information is called data fusion.

In 1987, the Joint Director's of Laboratories (JDL) Subgroup developed the Data Fusion Lexicon [64] as an approach to refine data collected from various systems. The JDL data fusion model was designed to take data from any aspect of the world, like flight information or network traffic, and process it in a way that the output is more useful. The output is supposed to better estimate, predict, or assess the environment under observation. The Data Fusion Lexicon defined data fusion as a method to combine information from "single or multiple sources" in order to estimate a situation, but Steinberg et. al [50] further refined the definition to "the process of combining data to refine state estimates and predictions."

The JDL data fusion model has five levels of data processing. Steinberg et. al [50] summarized and interpreted the JDL as follows:

Level zero: Sub-Object Data Assignment Level zero is the stage where the data is collected. Sensors determine or predict what state objects are in and reports that value.

Level one: Object Assessment This stage correlates sensor alerts together to form event tracks. Event tracks include the combined data to describe an attack. State information can be used to make inferences about singles objects.

Level two: Situation Assessment After the state information about single objects are obtained, the objects can be compared an analyzed. Knowledge aggregated from multiple objects provides the understanding of the current situation.

Level three: Impact Assessment The impact assessment stage involves analyzing the data and determining how the data affects the states, actions, or entities in the environment. Particularly, for actions that are occurring or could occur, this assessment determines how they will affect the next state or environment.

Level four: Process Refinement This stage involves taking the information learned about the previous stages and adapting the system. This may involve updated repositories or profiles. This stage depends on machine learning algorithms.

This five-level system can be seen in the following figure developed and presented by the JDL Subgroup [64].

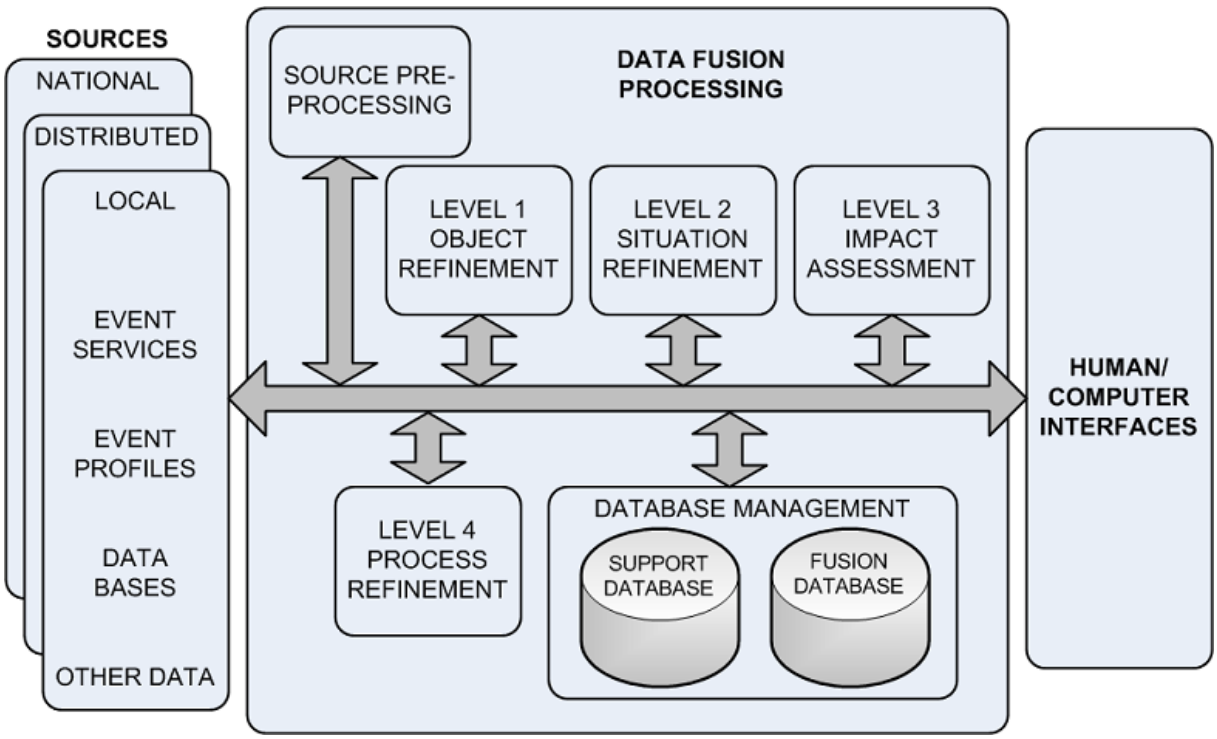

Figure 2.2: Original 1992 JDL Data Fusion Model

In the diagram, the sources and source preprocessing stages represent level zero. Following preprocessing, the data is refined and groomed in stages one through four. The data is then stored in a database and presented to the end user or additional systems.

Steinberg et. al [50] further proposed updates to the JDL data fusion model. They refined the framework to include trees and relationship graphs for how the information should be assessed by the system. In addition, they added additional explanations for each of the data processing levels on top of their interpretation of the original model.

An alternative data fusion and situational awareness model was developed by Dr. Mica Endsley [10]. Her approach was simpler but does not account for as much data as the JDL model. Endsley's model has three primary stages as seen in the following figure:

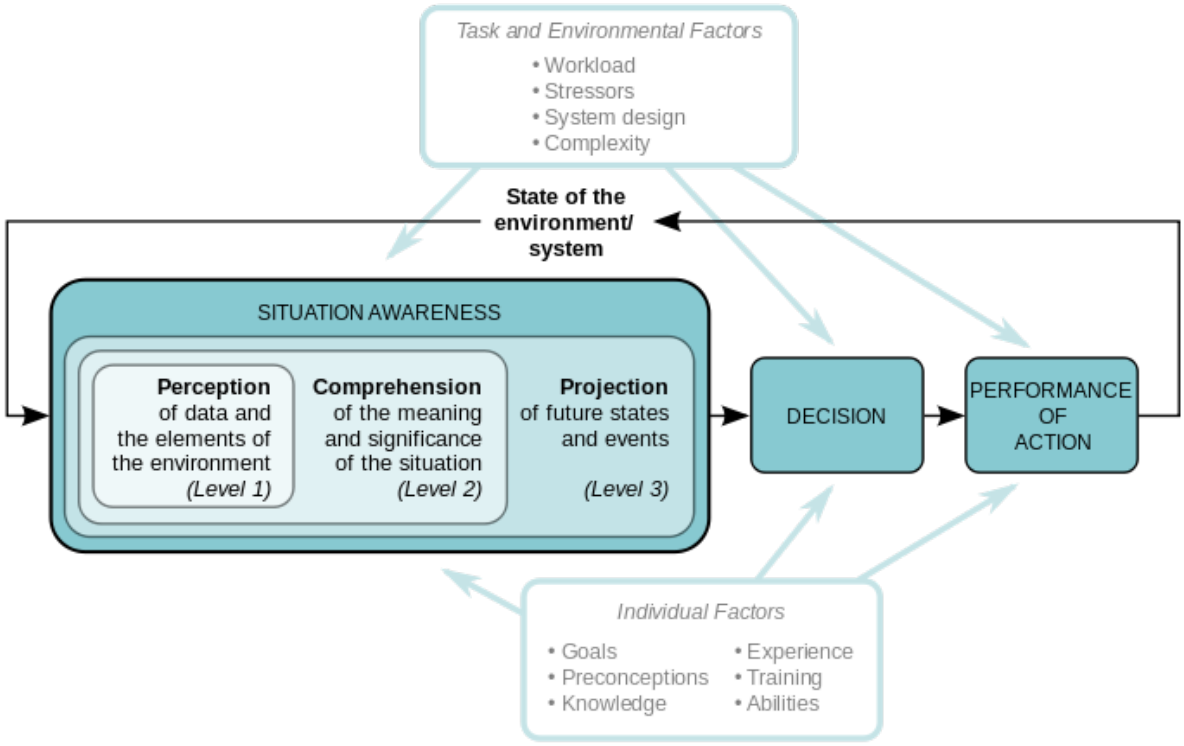

Figure 2.3: Endsley's dynamic decision making model

The first stage in Endsley's situational awareness model is similar to Level One in the JDL model, as it addresses having the information about a single object. Endsley's second stage resembles Level Two in the JDL model. The goal of the second stage is also to determine the current situation by assessing the relationships between objects. While the JDL model addresses both impact assessment and future state predictions, Endsley's model aggregates these into a single stage where state predictions are performed.

From these two competing frameworks, many branches in data fusion and situational awareness appear. One such branch by Wei Gao et al. [12] describe a data fusion approach which merges a rough set data with neural network data. They describe rough set theory as a method to decrease the data size by parsing through the data and removing redundant items. They claim it will potentially improve the speed and the identification rate within the overall system. Once their data set is reduced, they then aim on using learning algorithms to build a neural network. The end goal of their research is to improve training, or the learning phase of a neural network, before it is made operational.

Tran and Jin [61] proposed an approach that processed the same data set with a decision tree and a Gaussian mixture model. The two data outputs were designed not to overlap, but complemented on one another; one produced numerical data, while the other produced symbolic data. They performed two experiments, swapping the output type for the second experiment. Using this approach, there was a less than 1% improvement. The potential improvement of this research would be to use different processes that produce overlapping data, and then have a process that chooses the best output.

Network modeling research has previously been researched at Air Force Institute of Technology (AFIT). Lieutenant Ji 's research aimed to classify systems based on analyzing data from network sensors [18]. Her method of fusing data from both host IDSs and network IDSs resulted in a reduction of alerts. In addition, the data fusion showed greater effectiveness by about 28% versus individual classification systems. The output of

this research was limited to classifying systems as ”normal”, ”scanning”, and ”infected”. Lieutenant Ji ’s system ultimately did not provide enough information for cyber situational awareness, as it was limited by the state information employed.

Researchers at AFRL are conducting research on data fusion and cyber situational awareness in order to build a system resembling the JDL data fusion model. By combining the JDL data fusion model and Dr. Endsley’s dynamic decision making model, Tadda and Salerno [55] created a new model for situational awareness.

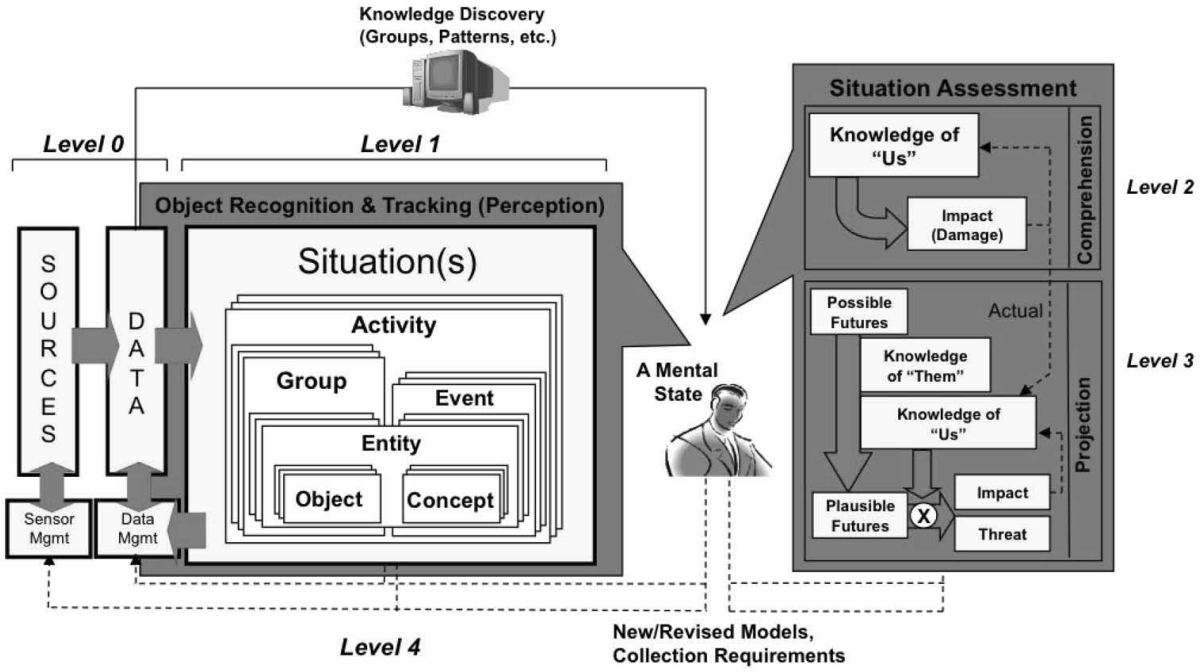

Figure 2.4: AFRL’s situation awareness reference model

In Tadda and Salerno’s situation awareness reference model, the original JDL model levels are split into two different processes. Levels zero and one, data acquisition and object correlation, are a single process. Within this object’s identification modules, they defined the object structures that they aim to track. In addition, JDL model levels two and three,

the situation and impact assessments, have been placed into another process devoted to situation assessment.

Through their studies, Blasch et al. [5], Salerno [41], and Tadda et al. [56] developed methods to measure the suitability of a system. Their metrics include system confidence, accuracy, timeliness, and relevance determinations on specific interests. These metrics focus on measuring the detection abilities of an IDS, the aggregation of IDS alerts, and the impact assessment system.

2.5 IDS Alert Aggregation

Raw data from intrusion detection systems are often overwhelming for administrators or systems to read and analyze due to the many alerts that are generated for each attack [25].

As some attacks occur in different stages, these alerts must be combined or clustered into single groups. Mathew et al. provide a method to associate IDS alerts into multistage attack scenarios [25]. This is accomplished through the process of template matching. As alert are processed through the system, their attributes are compared and matched against the template scenarios. Matching alerts are fit into ”scenario graphs” that represent the attacks. This method significantly reduces the amount of alerts that decision makers must analyze. In addition, the attacker’s goals can begin to appear as the ”scenario graphs” are analyzed.

Tadda et al. [56] explains that once enough observations are collected that portray a specific attack, the alerts can be aggregated into a track. They detail that tracks should represent steps in an attacker’s methodology. The tracks can then be managed over time by adding or removing observations and determining whether the attacks met their goals or failed. Tracks become a series of evidence that characterizes the attacker’s profile.

2.6 Impact Assessments

As a part of situational awareness identified in the JDL data fusion model [64], impact assessment is a critical process that lends to proper choices in defensive actions. Impact is defined as the ”force” or change that is applied from one entity to another [66]. Steinberg [50] interpreted the impact assessment phase of JDL data fusion model as as step to determining the ”effects” of events on objects. In cyber networks, impact can be viewed as the amount of damage or loss of a network entity from some event.

Researchers have looked at different approaches to accurately assess impacts on networks from cyber attacks. One technique by Argauer and Yang [2] aimed to perform impact assessments on services, users, individual hosts, and the network subnets. They then aggregated the values for a mean value for the attack. They used Common Vulnerability Scoring System (CVSS) values for the exploits in their experiment. The research also identified which attacks made sense and those that did not. The experiments were only performed on a simulated network environment with simulated attacks. The research did not specifically explain predictions for whether attacks were expected to succeed or fail. While the impact assessments accounted for impacts against the targeted hosts during attacks, it did not predict impacts against other hosts on the networks if the attacks were shifted to those other machines.

The aforementioned CVSS [27] is a system that provides scores to the vulnerabilities and attacks catalogued in the CVE online repository [29]. The scores depend on metrics such as knowledge that the exploit exists, the complexity of performing the attack, how many machines on the network would be affected, and how much access to the network the attacker requires to perform the attack. Additionally, the score depends on the exploits potential impact to target’s confidentiality, integrity, and availability. General scores are provided to exploits in the CVE catalogue, but using the CVSS calculator [32], custom scores can be computed by network administrators.

Salerno et al. [42] further refine the impact assessment process by identifying requirements needed for a true assessment to occur. They specify that the process requires both information about the attacker and information about the target network. The attacker's information includes what the attacker is capable of doing and what are the attacker's goals. The target or defender's information is necessary to determine if they attack can succeed. This information includes an inventory of the target network and vulnerability assessments for the network nodes. Comparing data from both perspectives gives insight to what effects actions will have on future states.

2.7 Uses of Cyber Situational Awareness

The primary reason for network situational awareness is so that defensive actions can be performed and that the network can be optimized. There are different ways to go about responding to attacks, which will be explained in this section.

2.7.1 Scripted Responses.

Several approaches have been used to respond to network intrusions. These approaches are far from fool-proof, but can be analyzed to determine how they can be improved through situational awareness.

The first technique is to automatically update firewall rules. Dr. Anton Chuvakin and Vladislav Myasnyankin [8] identified that the Guardian Active Response tool for Snort is an efficient method to provide an automated response to attacks. Guardian Active Response listens to the alerts generated by Snort to update the rules in real-time. The goal of Guardian Active Response is to prevent the action that caused the alert, eliminating future occurrences. Though, there can be issues where some alerts should be ignored or other defensive actions performed on the network for the sake of functionality. If legitimate traffic is not correctly classified, the legitimate traffic could be blacklisted along with the malicious traffic.

Another technique is called session sniping. Larsen [21] describes session sniping as a technique that begins when an IDS identifies a series of packets to be malicious. Since most of the packets already arrived at the host, they are outside of the firewall's ability to stop. With session snipping, a spoofed RESET packet can be sent to the host, telling the host to ignore those packets it already received. If the host had not yet processed a series of malicious packets, session sniping is useful to stop the traffic after it already arrived.

These techniques must be considered when developing a network situational awareness framework. The framework must be able to provide response tools the necessary information to formulate the best response. The two techniques above simply rely on alerts from an IDS or firewall, instead of a robust network picture, but there are tools that would require such a detailed situational awareness to efficiently operate.

2.7.2 Artificial Intelligence Systems.

The next method for automated cyber defense is to use a reasoning agent. Reasoning agents are made up of artificial intelligent (AI) systems may use the models to make decisions using game theory, plan future actions, and then learn from these actions. To build an efficient model to be used by AI agents, an understanding of these concepts and their requirements must be brought to light. The following research and definitions explain these concepts.

In AI, game theory is a view that multiple players in an environment compete against one another as if playing a game [39]. For this network defense environment, the player is the automated network defender, while its opponents are adversaries that attempt to attack the network.

Typically, an agent uses some type of search to determine the best courses of action in game theory [39]. Searches involve determining a large set of possible future states and determining the best set of actions that it takes to reach a goal state. A state is a snapshot of information about an object. To determine if some future state is better than another,

heuristics must be applied to the values within states. A heuristic is the cost or value in reaching a state.

Several examples exist that involve applying game theory to cyber defense. In evaluating network nodes that were infected with malware, Khousani et al. [19] considered the heuristics to be service degradation, resources used maliciously, host damage, and host destruction. In addition, reference [67] measured network services in their own network security model. They analyzed a server's availability, response rate, throughput, number of connections, and system errors. Each were quantified and combined to form the metric for that network service. Combined, these measureable network characteristics appear to be able to portray network health.

Additionally, Lye and Wing [23] advise that states should be measured with costs and rewards. Costs include resources, including time, which is needed to overcome a hurdle in the network or repair damage. Costs are denoted by negative values. On the other hand, rewards indicate positive values that are attractive to the attacker or administrator. A reward for the attacker may be the acquisition of information or the damage performed on the network.

Once states can be measured, AI systems can plan actions. Russell and Norvig [39] explain that planning in artificial intelligence is a method to find a set of actions to achieve a goal. This is accomplished determining the best sequence of states that map to a goal state.

State modeling and heuristic value determination help AI systems in analyzing information and making decisions. Providing the network information that applies to these described heuristics would be necessary when developing a situational awareness system. The right features of states must be included in the models that would support AI's determinations for which states are better.

2.8 Conclusion

From this review of related work, it is apparent that there is a strong emphasis on identifying and preventing cyber attacks. Many tools support this endeavor, but there needs to be a consolidation of these tools. There should not need to be redundant work, but new methods could prove useful that supports collaboration between tools and techniques. It is necessary for a data fusion method to exist between network intrusion detection systems and host intrusion detection systems. Information from these merged systems should have data filtered to represent only the most relevant information necessary to portray a computer network, while identifying cyber attacks and assessing their potential impact.

III. System Design

3.1 Overview

To create the prototype system and test the hypothesis, a monitoring system will be developed primarily from Commercial-Off-The-Shelf (COTS) software. The COTS products used will be responsible for the detection of cyber events. This research is limited to fusing the data from those sensors. If an event is not observed, it is at the fault of the COTS product. With COTS products used as the sensors of the system, the system will be designed for interoperability with other COTS products. For this experiment, the COTS sensors will be used on a small test network. The network will be sandboxed to allow a safe environment to perform malicious attacks that the sensors can observe.

In terms of data flow for the system, the sensors first observe network events, which are then logged to the database. Three analysis modules will query that database and collaborate on a single model in eXtensible Markup Language (XML) format. The model then can be used by decision-making entities to make a decision on how to administer the network. This process is depicted in the diagram on the next page.

When comparing the preceding data flow to John Boyd's Observe, Orient, Decide, and Act (OODA) loop [7], it is apparent that this process can be dividing into the four distinct actions in the loop: the sensors act as the observe phase; the modeler, action set identifier, and the attack criticality identifier act as the orient phase; the visualization and decision-making agents make up the decide phase; and the responder module suffices for the act phase. This research is more concerned with the ”observe” and ”orient” phases of the loop. This data flow is supported by other explanations of using the OODA loop in the cyber domain [43] [19], which were explained in chapter two.

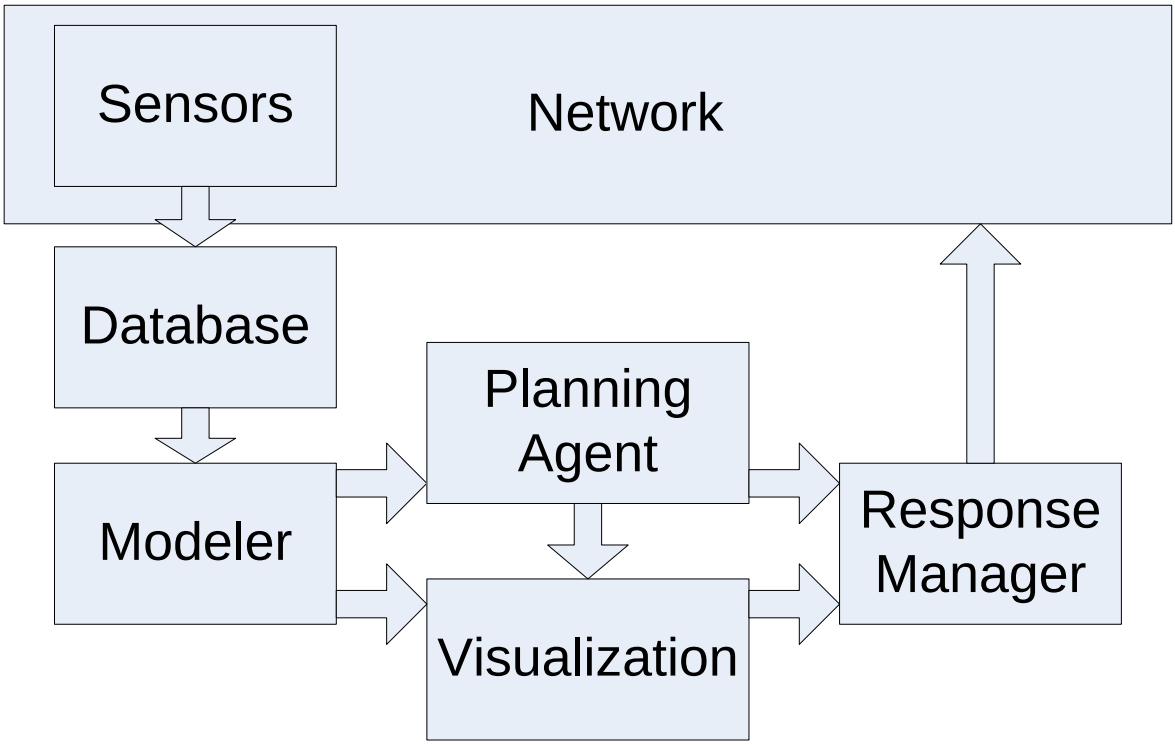

Figure 3.1: System Data Flow Diagram

3.2 Requirements

To provide cyber situational awareness to visualization and decision-making tools, the following requirements must be implemented into the program.

- The program shall mostly depend on COTS software for data acquisition.
- The program shall acquire host information, to include bandwidth, memory, and user information.
- The program shall reduce information by correlating Intrusion Detection System (IDS) alerts from the same attack into a single track or object.
- The program shall indentify the severity of impact an attack has against its target.

- The program shall report the model in XML format.

3.3 Approach

The following data flow diagram details how situational awareness will be achieved by the modeler program.

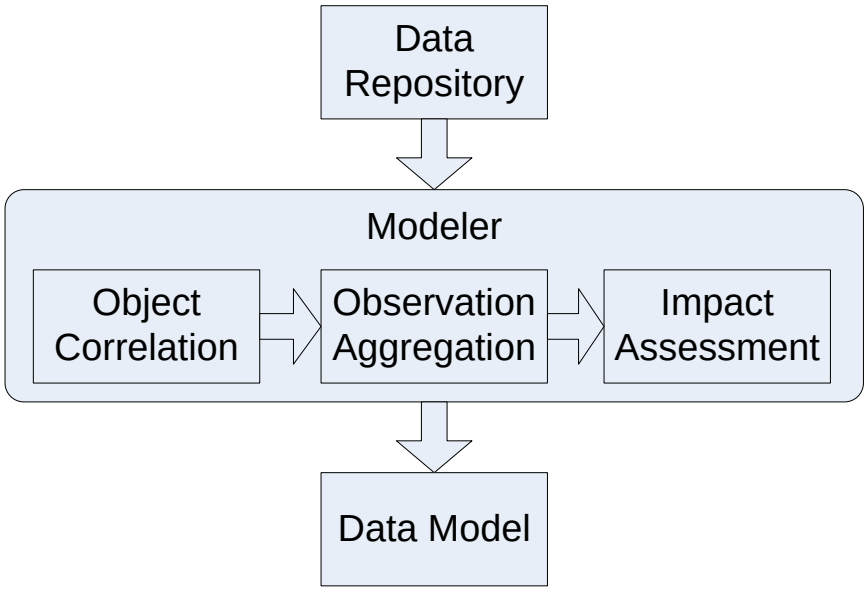

Figure 3.2: Modeler Data Flow Diagram

3.4 Design

The following subsections explain what COTS sensors will be used, how they will submit their data,

3.4.1 System Framework.

The following list of components and objects describes the information flow through the system.

Sensors The sensors shall be made up heterogeneous network filters, monitors, and scanners. Using sensors that acquire different types of data establishes a framework

for providing a broader awareness of the network. Some sensors may be installed on monitored hosts across the network, sending updates to the monitor. Other sensors will be installed on the monitor host, observing the network hosts.

Data store The database shall be the link between the sensors and the orientation components. Each sensor will insert records into a shared database as observations are made. The database will be installed on the network monitoring host.

Modeler The network modeling program shall query the database continuously for updates. The modeler will aggregate the network data and presented in a universally readable format, XML. The network will be the root object in the model; it will have child tags, such as hosts, the IDS, routers, and switches.

Action set identifier Once the XML file is created, the action set identification module shall identify the potential actions an administrator can perform on the network to defend against cyber threats. The module will query the database for actions based on the attributes found in the XML model. The actions may be different depending on the host, for instance, a user's account password may be reset on a client host, rules updated on the firewall, or the mail server receives a system update. Within the host tags, the available set of actions specific to that host will be listed.

Attack criticality identifier The attack criticality identification module shall determine the threat an observed attack has on its target and the other network hosts; it will provide a score that outlines the potential severity of the attack. This rating scale ranges from a one, at which the attack is expected to fail, up to a seven, where the attack is expected to succeed and the target suffer a high-degree of compromise. Section 3.7 outlines the rating system in detail. This score will be tagged and appended to its IDS event in the XML model.

XML model After the three modules complete their analysis of the network, the final XML model will be ready for analysis by decision-making agents.

Visualization and decision-making agents These are the decision-making components of the system, whether by human administrator or an artificially intelligent agent. This phase of the system is outside of the scope of this specific research, but will be modeled for completeness. For the experiment, an expert system will receive the network model and make a decision from a conditional tree of choices. The expert system will choose the action it deems as the best-choice from the action set in the model. Optimally, the agent should learn and develop a plan to mitigate the attack, but this experiment will simply use predetermined responses. When the best action is chosen, the expert system will inform the responder program.

Responder The responder program shall update the network per the instructions it receives from the decision-making agent. It should receive the action to be performed and the target host to perform the action on. The responder will connect to the target host via Secure Socket Host (SSH) protocol and send the instructions. The responder will inform to the decision-making agent with the result of the action, being either success or failure. The changes to the network should then be observed by the sensors and included in subsequent network models.

3.4.2 Data acquisition.

As the network changes, it is necessary to capture the new data for the battlespace model. These network attributes and events are observed by the system's sensors. The following subsections detail the network configuration, as well as the sensors used to observe the hosts and the events during experiment.

3.4.3 Sensors Selection.

The sensors that will be installed on the network will consist of a filter, a monitor, and two scanners. Since these sensors are of different types, they are considered heterogeneous of each other. Data from similar sensors will yield only a narrow view of the network. Research has shown that using multiple sensors of the same type will provide more accurate alerts [16] [58], but for this prototype, single monitors of a specific type will be sufficient. The goal is to show that multiple dissimilar sensors can provide the data necessary for a battlespace picture.

An Ubuntu Desktop Linux distribution in a VMWare Virtual Machine (VM) will be used to host the sensors and database. This virtual machine will act as the sensor for the network. In a production system, a system implemented and used by the community or select end users, this configuration may not be optimal since it provides a single point of failure. This configuration is suitable for this prototype system, improvements must be made for this system to be scaled up for real networks, which is discussed in chapter six.

The following diagram shows the data flow from the various network sensors to a single repository. The necessary situational awareness needed to model a cyberspace network depends on all of these sources.

The following subsections describe the chosen sensors for the system and how the sensors are implemented in the sensor host and used across the network.

3.4.3.1 Snort.

Snort is a very popular firewall and IDS [49] and will be used in this network to watch the network traffic for specific signatures. It will be installed on the sensor host and will serve as the gateway between the external and internal networks. It will sniff network traffic and log specific events to the database when observed. For the internal network, it will be able to sniff traffic that passes between hosts. This is allowed when the network switch is set to promiscuous mode and forwards copies of all packets to the sensor machine.

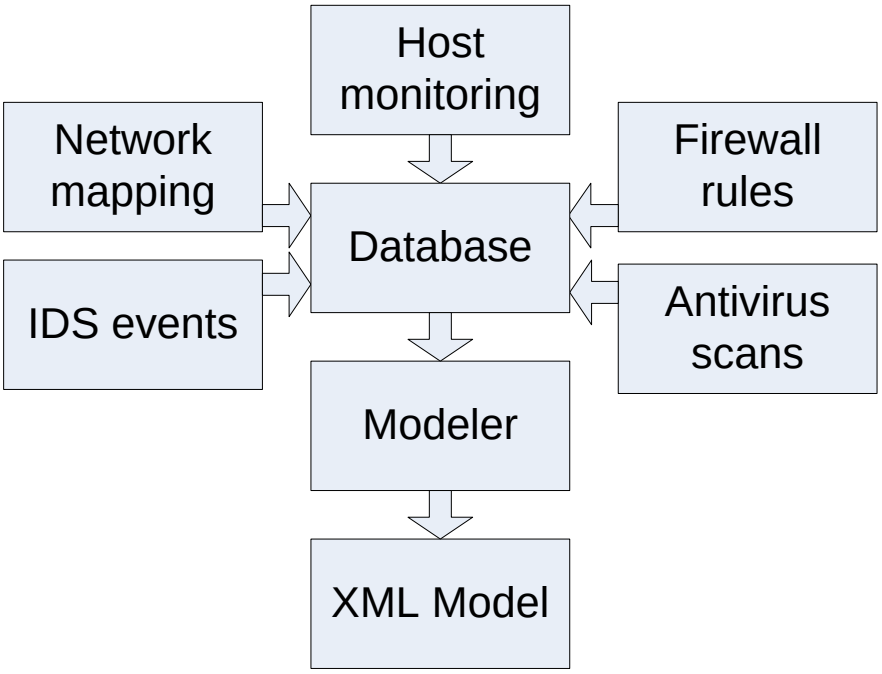

Figure 3.3: Sensors Input Data Flow Diagram

Snort will be the primary tool in classifying attacks on the system. It contains a large rule set that allows it to identify the traffic type and correlate it with a Common Vulnerabilities and Exposures (CVE) number. This will be included in the event logs and used by the other components in the system for modeling and rating the attacks.

The most critical configuration files for Snort is the Snort configuration, snort.conf, and the Snort rules, found in the rules folder. The snort.conf file indicates which rules files to use. In this system, all of the rules were disabled except for local.rules, which are supposed to be the network administrator generated rules for their specific system. By disabling all of Snort's default rules, this system can focus on identifying and correlated a specific set of rules which allows better accuracy in measuring the system.

The rules in ”local.rules” that are used to identify the attacks are detailed below. Most of the references were updated so that the modeler program can correlate the attacks to system vulnerabilities.

To identify Netcat attempts, the following rule was developed that caught the command shell string in the packet from the Netcat listener to the attacker.

```
alert tcp $HOME_NET any -> abt any (msg:"TELNET Netcat shell
exploit"; flow:established,from_server; content:"|6e 63 31 31
31 6e 74 3e|"; rawbytes; classtype:shellcode-detect;
reference:afit,11; sid:7701; rev1;)
```

To identify Structured Query Language (SQL) injection attempts, Nmap reconnaissance scans, File Transfer Protocol (FTP) password guessing, and SYN Flood denial of service attacks, the following rules were used from the default set of Snort rules.

```
alert tcp any any -> $HOME_NET 80 (msg:"SQL injection attempt";
flow:established,to_server; content:"|3d 27 31 3d|"; rawbytes;
classtype:web-application-attack; reference:afit,08; sid:8642212;
rev:4;)
```

```
alert tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"ET SCAN Potential
FTP Brute-Force attempt"; flow:established,from_server; content:
"530 "; pcre:"/530\s+(Login|User|Failed|Not)/smi"; classtype:
unsuccessful-user; detection_filter: track by_src, count 10, seconds
5; reference:afit,01; sid:2002383; rev:10;)
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (flags: S; msg:
"Possible TCP SYN Flood DoS"; flow:stateless; detection_filter:
track by_dst, count 60, seconds 5; reference:cve,1999-0116;
sid:10001; rev:01;)
```

```
alert tcp $ HOME _NET any -> $HOME_NET any (msg:"ET SCAN NMAP -sS
window 1024"; fragbits:!D; dsize:0 flags:S,12; ack:0; window:1024;
detection_filter: track by_dst, count 10, seconds 5; reference:
afit,04; classtype:attempted-recon; sid:2009583; rev:2;)
```

The following rule from EmergingThreats.net [6] is one of many created to catch the NETBIOS SMB buffer overflow that is implemented by Metasploit. This specific rule triggered the alert during development.

```
alert tcp any any -> $HOME_NET 445 (msg:"ET NETBIOS Microsoft
Windows NETAPI Stack Overflow Inbound  - MS08-067 (15)"; flow:
established,to_server; content:"|1F 00|"; content:"|C8 4F 32
4B 70 16 D3 01 12 78 5A 47 BF 6E E1 88|"; content"|00 2E 00
2E 5C 00 2E 00 2E 00 5C|"; reference: url, www.microsoft.com/
technet/security/Bulletin/MS08-067.mspx; reference:cve,2008-4250;
reference: url, www.kb.cert.org/vuls/id/827267; reference:url,
doc.emergingthreats.net/bin/view/Main/2008705;  classtype:
attempted-admin; sid:2008705; rev:5;)
```

To identify email account mail overflow attempts, the following rule was crafted. It performs similarly to identifying denial of service attacks, but it focuses on port 25 traffic.

```
alert tcp any any -> $HOME_NET 25 (msg:"POP3 Mailbox overflow";
flow:established,to_server; metadata:service pop3;
detection_filter: track by_src, count 100, seconds 5;
reference:afit,02; sid:963259; rev:19;)
```

3.4.3.2 Nmap.

Nmap is a network exploration tool that will be used to scan the subnet for hosts. Once it discovers live hosts, it can scan the hosts to determine their configurations. When used by malicious actors, a network scan is threatening, but when used by network administrators, it yields useful information.

Nmap will be installed on the sensor machine to scan the hosts on the network and determine the versions of their operating systems and what services are open. The results are saved to the database using a program called PBNJ [1], which is a set of Perl scripts. This data will help the modeling agent determine what vulnerabilities exist in the network.

A disadvantage of PBNJ is that it does not remove hosts from the database once the host is no longer on the network. This presented a problem in determining whether hosts were connected to the network or not. For instance, PBNJ could determine whether a rogue host appeared on a network, but could not portray in the database when it was no longer observed. To solve this, the scripts were updated to drop all records right before an update, so only the most recent scans were in the database.

3.4.3.3 Host Monitoring.

In addition to observing traffic and vulnerability scans, a method is needed to gather specific data from each host. A competent commercial-off-the-shelf tool is Nagios [30], which is a host health monitor that consists of a core server and distributed clients on the hosts. Nagios can provide information on Central Processing Unit (CPU) utilization, memory usage, running processes, and user information. This data is useful to identify unauthorized accounts and other abnormal behavior specific to host machines.

For the research network, installing Nagios became very difficult and many bugs appeared. Online guides support that stance that Nagios is difficult to set up, event for network experts [14]. To overcome this, a custom-built tool that is analogous to Nagios is developed for this research system. Written in Java, the tool resides on the desktop and server machines and acquires data using system commands and regular intervals. The host monitors then make a remote connection to the sensor machine's database to report the information.

Like Nagios, the custom host monitors acquire many critical values about the host. These values include memory usage, CPU usage, bandwidth tests, and various service states. In addition, the host monitor program runs the local anti-virus program, explained in the next section.

3.4.3.4 Antivirus.

To be able to observe the hosts' antivirus health, AVG Anti-Virus 2013 was installed on each desktop machine. It is used primarily because it is freeware and can be easily managed via command line. Other anti-virus programs should have the same features and can easily replace AVG in this system. The host monitoring program on each desktop machine initiates an AVG scan using command-line instructions. The results are then sent to the sensor machine.

Via command line, AVG can perform entire computer scans or scans of specific folders. This is useful when a virus is found on one computer; focused scans can be performed on the other hosts on the network. It is likely that the virus would be in the same directory, so scans of that directory on other hosts should yield whether the virus exists. These focused scans on single folders or directories are much quicker than complete computer scans.

Antivirus actions are normally automated within a host and results are confined to that machine. This system collects the data found and uses it to build a greater understanding of the local network.

Aggregating data from Snort, Nmap, the host monitoring programs, and AVG will build a view of the local network that is essential to defend in cyberspace. The data from these sensors are not redundant; instead, the data from each source complements each other. With the sensors in place, the system is postured to identify cyber attacks.

3.5 Data repository

As all of the sensors are gathering data, a central data repository is required to store the data prior to correlation by the modeling program. MySQL is chosen for this due to its simplicity and that it is freeware. In MySQL, four databases are used to store the various information: one for the Snort IDS, one for the mapping program PBNJ, one for vulnerability scores and requirements, and one for storing data collected from the host monitoring software and firewall. The following subsections describe how these different databases store data.

3.5.1 Snort database.

The Snort database is automatically configured during Snort's installation [49]. There are several tables of specific importance. The following figure shows the MySQL table list for Snort.

Table 3.1: Snort MySQL Tables

Tables in Snort
base_users
base_roles
data
detail
encoding
event
icmphdr
iphdr
opt
reference
reference_system
schema
sensor
sig_class
sig_reference
signature
tcphdr
udphdr

The ”event” table contains the primary information for alerts that are observed. This table contains the timestamp that the event occurs, a signature identifier that references an attack in the signature table, and the identifier for the signature that is associated with more data amongst the other tables.

The ”iphdr” table contains more information about an event. It has a foreign key that is associated to the event identifier in the ”event” table. This table contains data such as the source Internet Protocol (IP) address and port and the destination IP address and port.

The ”signature” table contains the full name for an attack. In addition, each signature has a reference identifier that points to a record in the ”sig_reference” table.

The ”sig_reference” table solely identifies the relationships between signatures and references.

The ”reference” table contains reference codes written in the Snort rules. For the modeling system developed in this research, this table has an additional use. Records in the

”exploit” table in the ”vuln” database references records in this table. This is explained in section 3.8.

3.5.2 PBNJ database.

The PBNJ database is the repository for the results PBNJ receives when it uses Nmap. The following figure shows the MySQL table list for PBNJ.

Table 3.2: PBNJ MySQL Tables

Tables in PBNJ
machines
services

The ”machines” table contains the information about discovered hosts by Nmap. It contains information such as the IP address, estimated Operating System (OS), and the time and data discovered.

The ”services” table holds the data for the services discovered in each host. Each service references the machine identifier for its corresponding host in the ”machines” table. The columns include service name, version, banner information, and whether the service is active or inactive.

3.5.3 Monitor database.

The next database is the ”monitor” database that store all the information acquired from the host monitoring software. The following figure shows the list of tables in the ”monitor” database.

The ”av_scan” table contains the results of antivirus scans. This record data includes the time that the scan occurred, the number of items scanned, and the number of high, medium, and low level infections. The actual infections are stored in the next table.

Table 3.3: Custom-built MySQL Tables

Tables in monitor
av_scan
host
infection
ip_address
resolved_event
user

The ”infection” table stores the data for discovered infections during antivirus scans. It contains the name of the malware, the type of malware, like worm, Trojan, or virus, and the location of the malware on the host.

The ”host” table contains most of the information about the host that the monitoring program acquires. This information includes hostname, OS name and version, network throughput estimates, memory, storage, and state information for various system services.

The ”ip_address” table contains the IP addresses discovered by the monitoring software. This data is in a separate table from ”host” since machines can have more than one IP address.

The ”user” table contains the usernames found local to the host that was discovered by the monitoring software. As with the IP address table, this data is in a separate table from the ”host” table as machines can have more than one user. For most machines, this data contains the local user accounts, but if the host is an Exchange server, then the user list is pulled from Active Directory and the users are domain accounts.

The final table that needs explaining is the ”resolved_event” table. This table was created to allow outside entities that are using the modeler’s output to specific attack information that they no longer want displayed. Normally, the attack track will remain in the model until it is acknowledged and dismissed. When administrators no longer require situational awareness of the track, their system or tools can update this table with the unique identifier of the alert in Snort. If the attack is not currently ongoing within the

network environment, like a persistent Denial-of-Service (DoS) attack, then this table tells the modeling software to ignore the attack track when creating the network model. The data will still be in the databases, but for the perspective of the users, the attack information will be absent.

3.5.4 Vuln database.

The final database in the system is the ”vuln” database. It contains three tables that store CVE reference data.

Table 3.4: Exploit MySQL Tables

Tables in vuln
exploit
vuln_os
vuln_svc

The ”exploit” table contains the name of the attack, the Common Vulnerability Scoring System (CVSS) score, which is explained in section 3.7. For attacks that appear in the ”snort” database, the signature correlates to exploits in this table. The system then checks the target host machine, using data from the ”pbnj” and ”monitor” tables, against the requirements in the next two tables.

The records in the ”vuln_os” table correspond to exploits in the ”exploit” table. This table contains the vulnerable OSs for the attack. If the target network matches the requirements in this table, the system assumes that the target is vulnerable against the attack.

Like the ”vuln_os” table, the records in the ”vuln_svc” table correspond to exploits in the ”exploit” table. This table contains the vulnerable services for the attack. The system assumes that the target is vulnerable against the attack if the target network matches the requirements in this table.

The database and table schemas outlined in these sections make up the complete data repository for the network's situational awareness. As the data appears in these databases, it is not easily interpreted by users. The modeling system, explained in the next few sections, will take this data and present it in an understandable format to users.

3.6 Primary data orientation process

While the monitors are observing the network and providing updates to the monitor's database, the modeling program will continuously query the database and create XML files. The resulting XML file is what shall be used to describe the cyber battlespace. The XML tags shall describe the network in terms of hosts, their services, statistics, antivirus scan results, and the IDS events of the network. The figure below shows the object hierarchy in the XML file.

```
<network>
<host>
[attributes]
<service>[attributes]</service>
<user>[attributes]</user>
<event>[attributes]</event>
<av_scan>
[attributes]
<infection>[attributes]</infection>
</av_scan>
<action>[attributes]</action>
<firewall_rule>[attributes]</firewall_rule>
</host>
</network>
```

Figure 3.4: Example XML Model

The ”network” tag is the root node. The ”host” objects are children of ”network”. All other objects, to include IDS events, services, antivirus information, users, and available

management actions are all children of ”host”. If the host machine is a firewall, it will have the ”firewall” rules.

To fill out the XML model, the following table shows where the data is collected for each set of attributes. This diverse conglomeration shows how the different data sources present data to be merged into a detailed image of the cyber network.

Table 3.5: Network Node Attribute Sources

Node Type	**Attribute**	**Source**
IDS	sensor_id	Snort IDS
	[event] timestamp	Snort IDS
	[event] source_ip	Snort IDS
	[event] dest_ip	Snort IDS
	[event] attack_type	Snort IDS
	[event] criticality	Criticality Assessment Process
	available_action	Action Set Identifier Process
Firewall	id	Nmap
	rule	FTP
	interface	Nmap
	available_action	Action Set Identifier Process
Client Host	host_name	Nmap
	ip_address	Nmap
	operating_system	Host Monitor
	cpu_utilization	Host Monitor
	[user] name	Host Monitor
	ids_event	Snort IDS
	[infection] timestamp	AVG Antivirus
	[infection] type	AVG Antivirus
	[infection] location	AVG Antivirus
	available_action	Action Set Identifier Process
Server Host	host_name	Nmap
	ip_address	Nmap
	operating_system	Host Monitor
	cpu_utilization	Host Monitor
	[user] name	Host Monitor
	[user] privilege	Host Monitor
	[user] last_login	Host Monitor
	ids_event	Snort IDS
	application_attributes	Host Monitor
	available_action	Action Set Identifier Process

3.7 Action set identification process

The next process in modeling the network provides the available set of actions for each network host. These are the actions that an administrator should be able to perform on the host if the administrator has access and permissions on the host. These lists of actions are necessary for administrators or decision-making agents to know their potential choices in mitigating network attacks.

As described in chapter two, actions occur as transitions between states. Thus, it is necessary to include actions in the model, proposing possible branches to the next model. In knowing the actions, future network states can be forecasted and predicting when competing against attacking agents, in the fashion of game theory.

In a computer network, the actions could be those from the network administrator, user, host operating system programs, or attackers. For this action set, the available actions will be limited to what a user or system administrator would be able to implement. The list will not be extensive, but only encompass enough actions to prove the functionality of this system. The following subsections details the proposed implementation of the attack criticality assessment process.

3.7.1 Process Mechanics.

Upon correlating the host data into objects from the databases, the action identification process will have the specifics of all the observed network hosts in the network. The action identification process will then query the database for actions available for specific host types. The process will append the action to the host in the model if the criterion is met. For instance, if PHP is installed on a server and specified in the model, then the action identifier will include PHP-specific actions with the host. The following data flow diagram illustrates this process:

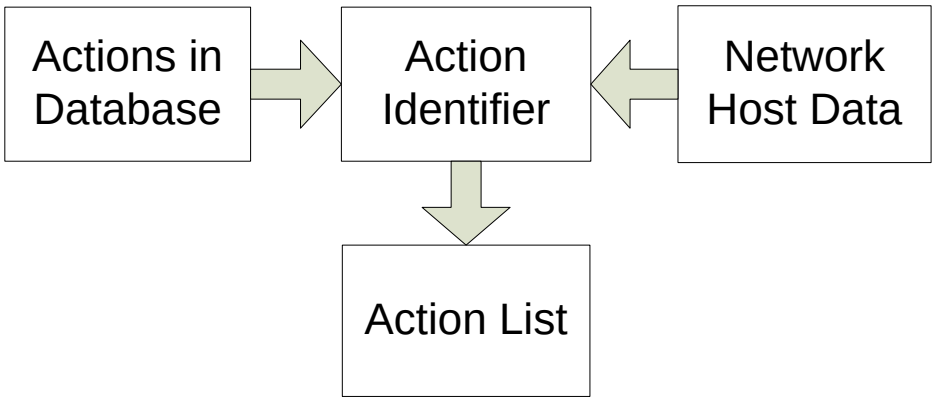

Figure 3.5: Action Identification Data Flow Diagram

For the experiment, the actions in the tables below will be appending to each host machine and firewall in the model. Routers actions will not be considered in this experiment, for simplicity. For actions that indicate association with a specific program, those will be appended to the host if the criteria are met; otherwise, the action will be added by default.

Table 3.6: Proposed Server Action Set List

Default Server Host Action Set List
Update operating system (if not latest version)
Update application (if not latest version)
Disable port/service
Enable port/service
Disable multi-line queries with PHP (PHP specific)
Enable multi-line queries with PHP (PHP specific)
Change user mailbox size (MS Exchange specific)

Table 3.7: Proposed Client Action Set List

Default Server Host Action Set List
Update operating system (if not latest version)
Update application (if not latest version)
Create user account
Delete user account
Reset user password
Disable port/service
Enable port/service

Table 3.8: Firewall Action Set List

Default Firewall Action Set
Block source IP address
Allow source IP address
Block destination IP address
Allow destination IP address
Block ICMP traffic
Allow ICMP traffic

3.7.2 Contribution of Action Set Identification.

When providing network administrators a set of available actions that can be performed on the network, disallowed options are stripped from the option set. This results in a quicker response and decreases the planning phase. When a decision-making agent performs a search of all possible states, a decrease in the action set will drastically decrease the search space.

A fielded implementation of action set definitions significantly improves modeling a cyber battlespace. For a fielded system, a complete list of actions should include all actions that an administrator can perform on the host, whether an action solves a problem or not. Optimally, the action set should include those that the attacker can perform; this expansive set of actions would give network defenders the foresight of possible future states that the network could reach.

3.8 Attack criticality classification process

The final process in analyzing and modeling the network state is aimed to rank the severity of the attacks on the network. The criticality assessment process determines if an attack possesses a threat to the network. It acquires the vulnerability data for the attack and determines if the target host or other network nodes meet the vulnerability criteria for the attack. Based off the analysis, the system assigns the attack a rating from ”1” to ”7”, with ”1” suggesting immunity and ”7” as having potentially disastrous consequences. The following subsections details the proposed implementation of the attack criticality assessment process.

3.8.1 Process Mechanics.

For timely assessment of an active threat, the system must first analyze the hosts on the local network. Tools such as Nessus or Nmap can determine what services are present on a host. Next, the system must have access to the vulnerability assessments from the CVE data store. These two sources will help the system assess the feasibility of a threat. Lastly, the system would require an IDS that can identify attacks entering or traversing through a local network. An attack can be observed by any number of intrusion detection systems, like Snort or Cisco’s Netflow, which can classify the attack.

Once the IDS identifies an event in the network, the assessment tool will use the attack type and the target of the attack. Based on the type of attack, the system will extract the host vulnerability requirements from the CVE database and compare it to the predetermined host data. From the comparison, a rating can be given to the attack in regards to the network.

The following data flow diagram shows the three input sources, the assessment system, and the output.

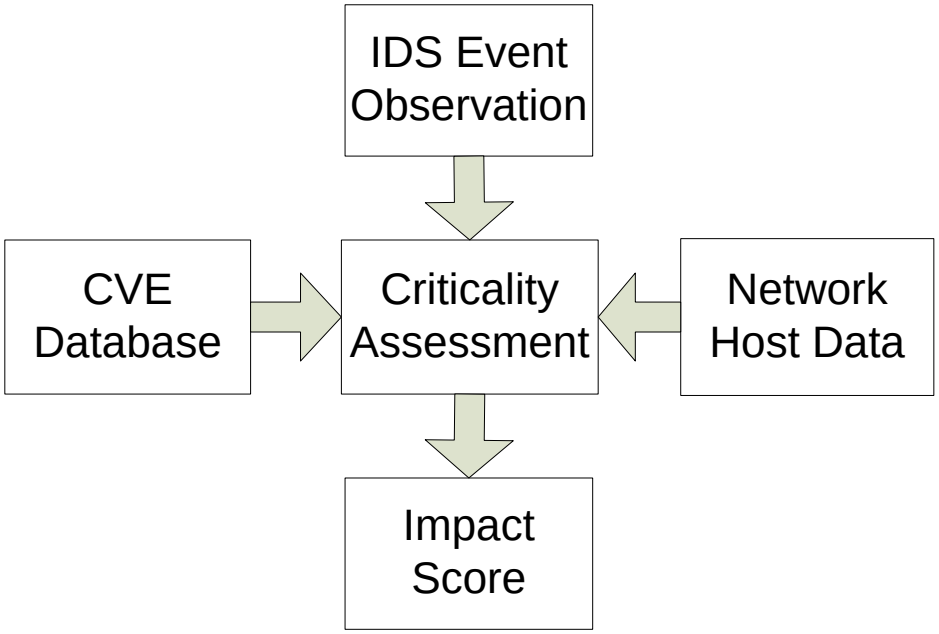

Figure 3.6: Impact Assessment Data Flow Diagram

3.8.2 Rating System.

As mentioned previously, this rating system is focused on the impact a specific attack has on a system. It accounts for the type of attack that is observed and its target. The scale correlates the attacks CVSS [27] with the expected outcome of the attack, a compromise or failure. In addition, the other hosts on the network are also accessed in terms of vulnerability.

For this scale, there are seven ranks: two for each of the high, medium, and low CVSSs, and a final rank signifying no expected threat. Of the two ranks in each CVSS category, the higher one signifies that the target is vulnerable, while the lower rank signifies expected immunity of the target but the existence of other vulnerable hosts on the network. This is important since upon failing an attack on one host, it is likely that the attack will be used against other hosts, as the attack is known to already be in the attacker's repertoire.

It should be noted that for this scale, it is more important to focus on a failing, high-level attack that could compromise the network in the very near future, instead of a moderate-level attack that is successfully exploiting a host. Per the CVSS assigned by CVE [29], the higher-level exploits are expected to do much more damage and it is normally wiser to mitigate those first.

Table 3.9: Threat Criticality Ranks

Impact	**Description**
7	Targeted hosts are vulnerable to the attack and high data loss and/or service degradation expected
6	The network possesses hosts that are vulnerable to the attack and high data loss and/or service degradation expected
5	Targeted hosts are vulnerable to the attack and moderate data loss and/or service degradation expected
4	The network possesses hosts that are vulnerable to the attack and moderate data loss and/or service degradation expected
3	Targeted hosts are vulnerable to the attack and minor data loss and/or service degradation expected
2	The network possesses hosts that are vulnerable to the attack and minor data loss and/or service degradation expected
1	An attack is observed, but all the network's hosts are expected to be resistant

The model does not discern the value of data on each network host. Once a host is exploited, is reasonable to assume that the attacker will attempt a new exploit on another network host by pivoting from the compromised host. Thus, the value of information should not be measured on a host-by-host basis, but at the network-level. Since this rating system is used within a single network and there exists the threat of a pivoting attack, the value of the data is considered equivalent between hosts. Though, the concern over confidentiality, integrity, and availability is not absent in this framework. The CVE score takes into account the confidentiality, integrity, and availability of the impact of the attack in general and is used to derive the level of this criticality rating.

3.8.3 Contribution of Criticality Rating.

A process for rating a specific attack against a specific host contributes to the cyber community. It builds upon the CVE system. The criticality rating can be used by an automated decision-making agent or a network administrator in determining the potential risk of an attack against their custodial network. Attacks can be prioritized, remedied, or ignored, depending on the rating itself and available corrective actions.

3.8.4 Example Execution of Criticality Assessment Process.

This section provides an example that elaborates on how data flows in the criticality assessment process and how scores are assigned and ranked. For this example, we identify two fictitious network hosts and a fictitious IDS identifies two plausible cyber attacks on the network. This example then explains how the scores are assigned to the threats.

For this example, assume a network contains only the two hosts in the following table:

Table 3.10: Example Vulnerable Hosts for Criticality Assessment Experiment

Example Id	**OS**	**Applications/Services**	**Version**
1	Mac OS X	VLC Media Player with UPnP Adobe Flash Player	1.0 10.3.9
2	Windows XP SP2	Universal Plug and Play (UPnP) Adobe Flash Player	SP2 10.3

For example, we could further assume that our IDS identified two simultaneous events with the CVE references CVE-2007-1204 and CVE-2012-4168. CVE-2007-1204 is targeting host ”1” and CVE-2012-4168 is targeting host ”2”.

For the two attacks, the following vulnerability descriptions are excerpts from the National Vulnerability Database (NVD) [32].

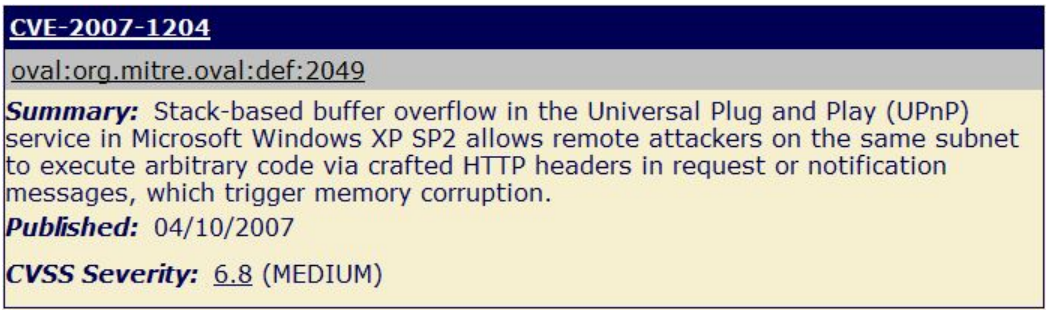

Figure 3.7: Snapshot CVE-2007-1204 from NVD

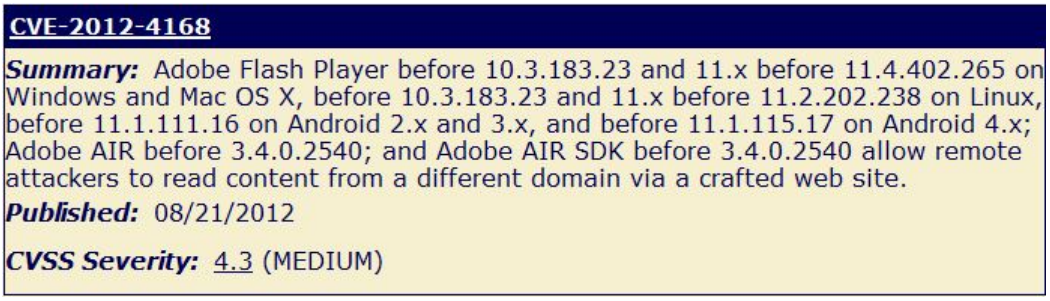

Figure 3.8: Snapshot CVE-2012-4168 from NVD

Upon receiving the IDS data and already having the network and CVE data, our criticality assessment tool would perform the following logic to assign ratings to the attacks:

1. Host ”1” (target) is not expected to be vulnerable to ”CVE-2007-1204” due to operating system mismatch.

2. Host ”2” (network resident) is expected to be vulnerable to ”CVE-2007-1204” due to operating system match and service/version match.

3. Since the target of the ”CVE-2007-1204” attack (MEDIUM severity) is expected to be immune but the network contains at least one host that may be vulnerable, the attack is assigned a criticality rating of ”4”.

4. Host ”1” (network resident) is expected to be vulnerable to ”CVE-2007-4168” due to operating system match and service/version match.

5. Host ”2” (target) is expected to be vulnerable to ”CVE-2007-4168” due to operating system match and service/version match.

6. Since the target of the ”CVE-2007-4168” attack (MEDIUM severity) is expected to be vulnerable, the attack is assigned a criticality rating of ”5”.

With this assessment of the two identified events in regards to the example network hosts, it would be advisable for defenders to correct or mitigate the attack, identified as ”CVE-2007-4168”, with a criticality rating of ”5” before attending to the attack, identified as ”CVE-2007-1204”, with a criticality rating of ”4”.

3.8.5 CVSS Scores and Predicted Impacts.

In designing this modeling system, a set of common attacks were chosen. These seven attacks include network scan, FTP login brute-force attempt, mail box overflow attempt, SYN Flood denial of service attack, Netcat backdoor connection, NETBIOS buffer overflow exploit, and an SQL webpage injection. Not all attacks are identifies in the CVE database. CVE recommends that organizations determine the severity of vulnerabilities specific to their own systems [29]. To determine severities, CVE provides the CVSS version 2 calculator [29] to compute the scores. The following values were used to determine the CVSS scores.

Table 3.11: CVSS calculation for FTP brute-force attempt

Metric	**Value**
Exploitability Metrics	
Access Vector	Network
Access Complexity	Low
Authentication	Single Instance
Impact Metrics	
Confidentiality Impact	Complete
Integrity Impact	Partial
Availability Impact	None
General Modifiers	
Collateral Damage Potential	High
Target Distribution	Medium
Impact Subscore Modifiers	
Confidentiality Requirement	High
Integrity Requirement	Medium
Availability Requirement	Low
Temporal Score Metrics	
Exploitability	High
Remediation Level	Temporary Fix
Report Confidence	Confirmed
Overall CVSS Score	**6.8**

The calculator gave an FTP brute-force attack a score of ”6.8”, which makes it a medium threat. Per the Attack-to-host impact assessment scale, Figure 3.9, this attack would have an impact of ”5” to a vulnerable system, an impact of ”4” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

Table 3.12: CVSS calculation for Netcat connection

Metric	**Value**
Exploitability Metrics	
Access Vector	Local
Access Complexity	Medium
Authentication	Single Instance
Impact Metrics	
Confidentiality Impact	Complete
Integrity Impact	Complete
Availability Impact	None
General Modifiers	
Collateral Damage Potential	High
Target Distribution	High
Impact Subscore Modifiers	
Confidentiality Requirement	High
Integrity Requirement	High
Availability Requirement	Low
Temporal Score Metrics	
Exploitability	High
Remediation Level	Temporary Fix
Report Confidence	Confirmed
Overall CVSS Score	**8.0**

The calculator determined that a Netcat connection would be scored at ”8.0” with the values provide. This score makes it a high threat. Per the Attack-to-host impact assessment scale, Figure 3.10, this attack would have an impact of ”7” to a vulnerable system, an impact of ”6” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

Table 3.13: CVSS calculation for email blitz

Metric	**Value**
Exploitability Metrics	
Access Vector	Network
Access Complexity	Low
Authentication	Single Instance
Impact Metrics	
Confidentiality Impact	None
Integrity Impact	None
Availability Impact	Partial
General Modifiers	
Collateral Damage Potential	Low
Target Distribution	Medium
Impact Subscore Modifiers	
Confidentiality Requirement	Low
Integrity Requirement	Low
Availability Requirement	High
Temporal Score Metrics	
Exploitability	Functional exploit exists
Remediation Level	Workaround
Report Confidence	Confirmed
Overall CVSS Score	**3.8**

The calculator gave the mail overflow attack a score of ”3.8”, which makes it a low threat. Per the Attack-to-host impact assessment scale, Figure 3.11, this attack would have an impact of ”3” to a vulnerable system, an impact of ”2” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

Table 3.14: CVSS calculation for NETBIOS SMB buffer overflow

Metric	**Value**
Exploitability Metrics	
Access Vector	Local
Access Complexity	Low
Authentication	None
Impact Metrics	
Confidentiality Impact	Complete
Integrity Impact	Complete
Availability Impact	Partial
General Modifiers	
Collateral Damage Potential	High
Target Distribution	High
Impact Subscore Modifiers	
Confidentiality Requirement	High
Integrity Requirement	High
Availability Requirement	Low
Temporal Score Metrics	
Exploitability	High
Remediation Level	Official Fix
Report Confidence	Confirmed
Overall CVSS Score	**8.1**

The calculator gave the NETBIOS buffer overflow attempt a score of ”8.1”, which makes it a high threat. Per the Attack-to-host impact assessment scale, Figure 3.12, this attack would have an impact of ”7” to a vulnerable system, an impact of ”6” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

Table 3.15: CVSS calculation for Nmap scan

Metric	**Value**
Exploitability Metrics	
Access Vector	Local
Access Complexity	Low
Authentication	None
Impact Metrics	
Confidentiality Impact	Partial
Integrity Impact	None
Availability Impact	None
General Modifiers	
Collateral Damage Potential	Low-Medium
Target Distribution	High
Impact Subscore Modifiers	
Confidentiality Requirement	Low
Integrity Requirement	Not Defined
Availability Requirement	Not Defined
Temporal Score Metrics	
Exploitability	High
Remediation Level	Workaround
Report Confidence	Confirmed
Overall CVSS Score	**3.7**

The calculator gave a network scanning a score of ”3.7”, which makes it a low threat. Per the Attack-to-host impact assessment scale, Figure 3.13, this attack would have an impact of ”3” to a vulnerable system, an impact of ”2” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

Table 3.16: CVSS calculation for SQL injection attempt

Metric	**Value**
Exploitability Metrics	
Access Vector	Network
Access Complexity	Low
Authentication	None
Impact Metrics	
Confidentiality Impact	Complete
Integrity Impact	Complete
Availability Impact	None
General Modifiers	
Collateral Damage Potential	Medium-High
Target Distribution	Medium
Impact Subscore Modifiers	
Confidentiality Requirement	High
Integrity Requirement	Medium
Availability Requirement	Not Defined
Temporal Score Metrics	
Exploitability	High
Remediation Level	Official Fix
Report Confidence	Confirmed
Overall CVSS Score	**6.9**

The calculator gave the SQL Injection attempt a score of ”6.9”, which makes it a medium threat. Per the Attack-to-host impact assessment scale, Figure 3.14, this attack would have an impact of ”5” to a vulnerable system, an impact of ”4” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

Table 3.17: CVSS calculation for SYN flood DoS

Metric	**Value**
Exploitability Metrics	
Access Vector	Network
Access Complexity	Low
Authentication	None
Impact Metrics	
Confidentiality Impact	None
Integrity Impact	None
Availability Impact	Complete
General Modifiers	
Collateral Damage Potential	Low-Medium
Target Distribution	Medium
Impact Subscore Modifiers	
Confidentiality Requirement	Not Defined
Integrity Requirement	Not Defined
Availability Requirement	Medium
Temporal Score Metrics	
Exploitability	High
Remediation Level	Unavailable
Report Confidence	Confirmed
Overall CVSS Score	**6.3**

The calculator gave a web server denial-of-service attack a score of ”6.3”, which makes it a medium threat. Per the Attack-to-host impact assessment scale, Figure 3.15, this attack would have an impact of ”5” to a vulnerable system, an impact of ”4” to an immune system with vulnerable neighbors, and an impact of ”1” to an immune network.

3.9 Modeler execution

To operate this system in a network environment, several components must be started. First the sensors should all be reporting. Snort must be running and logging events to the database. Host monitoring systems on the dispersed hosts should be running and remotely sending information to the database. The host monitoring software will call execute AVG antivirus program from command line and report on its behalf. PBNJ will perform Nmap scans of the entire network. In Linux systems, PBNJ can be added to the ”cron” service,

which automatically performs command per a schedule; this allows PBNJ with Nmap to run in the system background. Finally, the firewall rules are required from the firewall VM. Bash scripts performing FTP commands can pull the firewall's configuration file at regular intervals to check for changes and update the database.

With the sensors running, the modeler can be executed. It fetches the data from the database, correlates the information, then exports an XML file to the local web server. For this system, it was easier for the artificial intelligent (AI) agent and visualization network controllers to pull the model from the web server. Ultimately this is not secure, as it also can give attackers situational awareness of the network. This problem is discussed in chapter six.

Though options, the modeler will continually check for new data in the database and post new models at regular intervals. The length of time of these intervals can be specified in milliseconds.

3.10 Summary

This chapter explained how the cyber situational awareness modeling system was built. It detailed what sensors were used and how they were configured. The databases that store all of the network information were outlined. In addition, this chapter described the processes that correlate the data and provides impact assessments. All of these techniques come together to produce real-time snapshots of a network's situation.

The next chapter will explain how modeler is measured in terms of accuracy, data reduction, and the relevance of its processes. It will detail the methodology of the tests that were performed.

IV. Methodology

4.1 Introduction

This chapter explains the methods used to develop and test the system in its ability to perform data fusion and provide situational awareness. First of all, it defines the problem, explains the goals, and expresses the overall approach to the problem. Next, this chapter details how the system is configured and explains the attacks that will be used to assess the system. Finally, this chapter discusses the performance metrics and the experimental procedures.

4.1.1 Problem Statement.

As government and commercial entities become more dependent on their computer networks, these networks become a greater target for adversaries. To defend against cyber attacks, defensive measures must occur during the attack, or even immediately afterwards, in order to provide an appropriate response. For network defenders to be able to respond in a timely manner, they must have a thorough situational awareness of their cyber battlespace. Current situational awareness methods are insufficient [36] [24]. Network administrators and their tools require more details of the cyber battlespace to be able to compete and achieve the goals of their organizations.

For network analysis, the administrators possess too much data to dissect effectively to determine the actual threat of an attack and respond before it achieves its peak damage [36] [24]. To solve this problem, an automated process must exist to collect network event data and orients the data so that it is useful for analysis, leveraging commercial-off-the-shelf sensors. Much research [12] [22] [58] has been accomplished that fuses data of homogeneous sensors, like the observations of multiple intrusion detection systems. Though, aggregation of homogeneous sensor data only increases accuracy of traffic

identification. To obtain the right data that gives defenders the appropriate situational awareness, heterogeneous sensor data fusion must be present.

For the problems of data overload and the need for situational awareness, methodologies and frameworks exist, but the research community requires a simple approach or tool. This would allow researchers to accelerate their research by taking an existing framework and enable them to build upon it.

4.1.2 Goals and Hypothesis.

The goal of this research is to show that aggregated sensor data can be used to give network defenders the information needed while reducing irrelevant information. This experiment will test the efficiency and value added by uses the System Under Test (SUT). The SUT is the data fusion and situational awareness modeling system specified in chapter three.

The SUT was designed to gather and correlate information from five types of sensors: an Intrusion Detection System (IDS), host monitors, anti-virus software, a vulnerability scanner, and file transfers from the firewall. It is obvious that this system offers an enormous amount of testing opportunities, but per the limits of time, experimentation will be scoped to specific aspects of the system. Specifically, the experiment will analyze the system's ability to identify malicious traffic through the IDS alerts, reduce the alert volume by correlating the alerts into single related objects, or tracks, and provide impact assessments on how severe an attack my affect the network. The data under analysis results from the orientation and correlation of information acquired from the IDS and vulnerability scanner. The data from host monitoring and antivirus scans, explained in chapter three, will not be analyzed during this experiment. SUT for this experiment is specifically composed of the modeling program, the Snort IDS, and PBNJ with Nmap.

For an estimation of performance that the amount of data alert traffic will be reduced by two-thirds. This means that for every three alerts found by the IDS, only one track

will represent the attack. In addition, the accuracy and recall rate of the system should exceed 50%, which is randomness. This measurement will take into account both false positives, like fragmented tracks and misassociated tracks, and false negatives, like missing observations.

For the impact awareness process, the assignment of impacts should be completely accurate based on the signature and target of the attack. Measuring the accuracy of assigned impact will provide little value for this experiment. Instead, the worth of having the impact assessment as part of the system will be measured. Analysis will discover how much information is relevant and irrelevant to administrators if they are only concerned with a specific type of traffic. Since, the system provides impact assessments in real-time, any information determine irrelevant during the test can be observed as a reduction in workload as they operator would have the choice to ignore irrelevant information.

If administrators are only concerned with high-level impacts, it can be estimated that workload will be reduced by 71%, as two-sevenths of the attacks should be high impact. Additionally, if administrators are only concerned with high and medium-level impacts, it can be estimated that workload will be reduced by 29%, as high and medium impact attacks make up five-sevenths of the projected attacks.

4.1.3 Approach.

To evaluate the SUT, it will be installed in a real network and test with real attacks. The environment will consist of a virtual network, explained in the next section.

This experiment will use real cyber attacks against the virtual network. The cyber attacks to be performed on the network are explained in section three.

The cyber attacks will occur at random during twenty-five tests. It is desired that each test have at least thirty attacks occur. An estimated 750 attacks should occur over the course of all twenty-five tests. These quantities should provide enough confidence to the measurements of the tests. The experiment itself will be outlined in section four.

Section five will detail the metrics applied to the data after the experiment completes. Next, section six will explain the limitations for this experiment. The final section will summarize the methodology used for this experiment.

4.2 System environment

The SUT requires a real network that to reside within and will act as the environment for the tests. This network will be virtualized on a server using VMWare ESXi as the hypervisor. The SUT will contain nine Virtual Machine (VM)s and two switches. The VMs include the sensor VM containing the SUT, attack targets, the black hat VM, and a firewall. The following table details the characteristics of the server that will support the entire environment:

Table 4.1: Server Configuration

Component	**Value**
Central Processing Unit (CPU) capacity	2 x 3.46 Gigahertz (GHz)
Memory capacity	192.0 Gigabyte (GB)
Disk capacity	1.8 Terabyte (TB)
Hypervisor	VMWare ESXi 5.0.0

According to the Dr. Eric Cole [9], the typical functions in modern information technology (IT) networks consist of web browsing, file sharing, electronic mail, voice services, and information security. The computer network for this experiment will provide these services so that it closely resembles common networks. Depending on the size of the network, services may be consolidated amongst the hosts. The environment is a small network compared to that of many corporations, but is more common for smaller businesses. The network's Microsoft Windows server will perform Exchange, Domain Name Service (DNS), and domain control services, while the Linux servers will operate as a web server and a File Transfer Protocol (FTP) server.

For the experiment, attackers will penetrate the network from the outside; the attacks may travel through the firewall to reach the internal network. The VM containing the SUT connects to a switch that services the hosts on the internal network. The switch sends copies of all network traffic it sees to the SUT so that the IDS can perform traffic sniffing.

Larger networks may create a demilitarized zone between the Internet and the internal network, buffered by firewalls on both sides. The SUT does not implement a DeMilitarized Zone (DMZ) because the network topology itself is not the focus of this research, but instead, the performance of the sensors and the modeling agent. Without a DMZ, this network is more vulnerable, but will make for simpler attack scenarios.

The following diagram shows the network topology of the environment. The black hat VM has both internal access directly to the switch and external access through the firewall. This allows the attacking VM to perform cyber attack from both vectors, realistic in real-world scenarios [9].

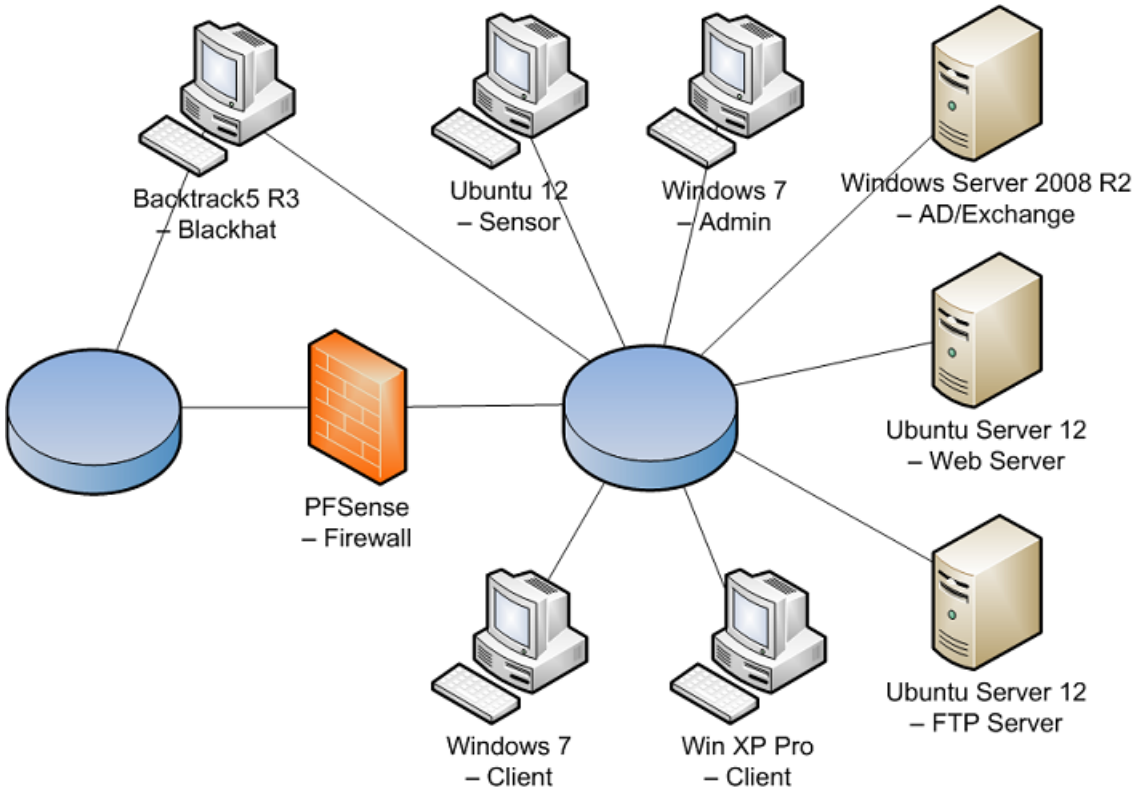

Figure 4.1: Virtual Network Topology

The following table details the configurations of the hosts. The target hosts have varied operating systems and applications that will contain vulnerabilities to a range of attacks, which will be discussed in section the next section. Machine one will contains the SUT, which will be observing the activities within the environment. The hosts identified as targets will be attacked during the experiment. Machine eight is the firewall; external attacks will pass through the firewall, which also makes it a target. Machine nine is the black hat VM that will be performing the attacks on the target hosts.

As seen in Figure 4.2, the black hat machine is buffered by the firewall for some attacks. The firewall will send the traffic to the email server, web server, or FTP server, depending on the destination port identified in the packers. The attacker is also given direct

Table 4.2: Configuration of VMware Guest Machines

Id	OS	Version	CPU	RAM	Extra Software	Role
1	Ubuntu Desktop 12	3.2.0-27-generic-pae	3.46 GHz	4 GB	Snort Nmap/PBNJ Apache MySQL	Sensor
2	Windows 7 Enterprise 32-bit		3.47 GHz	1 GB	WinSCP putty Host Monitor	Target
3	Windows 7 Enterprise 32-bit		3.47 GHz	1 GB	Microsoft Outlook Host Monitor AVG	Target
4	Windows XP Professional	SP2	3.47 GHz	512 MB	Microsoft Outlook Host Monitor AVG	Target
5	Ubuntu Server 12.04 LTS	3.2.0-23-generic-pae	3.46 GHz	1 GB	Apache MySQL PHP FTP Host Monitor	Target
6	Ubuntu Server 12.04 LTS	3.2.0-23-generic-pae	3.46 GHz	1 GB	Apache MySQL PHP FTP Host Monitor	Target
7	Windows Server 2008 R2 64-bit Enterprise	SP1	3.47 GHz	4 GB	Domain Controller Microsoft Exchange Host Monitor DNS	Target
8	PFSense	2.0.1	3.46 GHz	1 GB		Firewall
9	Backtrack5 R3	3.2.6	3.46 GHz	1 GB		Attacker

access to the internal network, bypassing the firewall. Depending on the attacks, some actions will be sent through the firewall, while others straight to the internal machines.

Prior to performing any attacks on the virtual network, it is necessary to save snapshots of each virtual machine and sandbox the environment. Sandboxing is accomplished by placing the hosts on a private network with no external access. This will ensure that malicious data cannot leak out and affect other network nodes. When an experiment is

completed, the virtual machines can be rolled back to a safe snapshot and a new experiment can be conducted.

4.3 Attack Traffic Selection

For the experiment, real cyber attacks will be used to test the network. These attacks will mostly be supported by the BackTrack5 [3] suite of tools. Some attacks may not require the tools, but will be performed from the BackTrack5 VM.

4.3.1 BackTrack5 and Metasploit Framework.

To remove unpredictability in the success of the attacks against the SUT, the black hat VM will contain specifically crafted attacks for the experimental network. The black hat VM with BackTrack5, machine nine, will be used for this purpose. BackTrack5 is a Linux Operating System (OS) distribution with a suite of hacking tools installed. One of the tools, the popular Metasploit Framework, will be used for one of the attacks.

The purpose of this separate set of attack traffic is that community data sets are not guaranteed to compromise the target. The crafted attacks from BackTrack5 will be developed specifically to exploit the hosts on the network. This assurance is necessary for the experiments so that compromised hosts are observed in the network model.

The following seven cyber attack scenarios will be crafted and implemented using the BackTrack5 tools suite:

4.3.1.1 Network mapping and port scanning.

The BackTrack5 VM shall execute an Nmap scan of targeted VMs. It will perform one scan against a single target at a time, since scans against multiple hosts can be time and resource exhaustive.

4.3.1.2 Structured Query Language (SQL) injection.

The web server, machine five, will have a PHP page containing a text field for searching a database. The PHP page will be vulnerable against SQL injection. It will

accept multiple queries in a single execution. The black hat VM will implement this attack against the web server to represent acquiring more data than the webpage intended.

4.3.1.3 OS exploit.

Using the Metasploit framework in the BackTrack5 VM, a Server Message Block (SMB) request buffer overflow will attempt to exploit one of the client machines at random. Upon success, a Meterpreter session will be established on the vulnerable VM. Only the Windows XP VM, machine four, is actually vulnerable. Though, the IDS should identify attempts against all three client machines.

4.3.1.4 Denial-of-Service (DoS).

An attempted DoS attack shall be implemented from the BackTrack5 VM. Upon execution, the BackTrack5 VM shall spawn a random number of DoS attempts, potentially resulting in an attempted Distributed Denial-of-Service (DDoS) attack. Since all of the DoS attempts are originating from the same VM, the severity of the attack will be low, but the Snort rule in the IDS will use a lower threshold to identify the attempt.

4.3.1.5 Password guessing.

Within the BackTrack5 suite, the password guessing program, Brutus, will send password guesses to either the web server or the FTP server. For this experiment, the attack is not expected to succeed within the time frame but the IDS should identify the attempt.

4.3.1.6 Mail Blitz.

Microsoft Exchange is installed on the Microsoft Windows Server VM. The BackTrack5 VM will spam known email accounts with hundreds of emails. This will represent an email blitz attack that aims to exhaust the storage space in the account, thus denying the target the ability to receive valid email traffic.

4.3.1.7 Netcat connection.

The Windows 7 client, machine three, will have a Netcat installed and persistently listening for commands. This will represent a compromised host with an installed backdoor. The BackTrack5 VM will make connections to the compromised host and establish command shells.

These attacks were chosen since they are representative of the various steps of cyber attack methodologies [48]. The Nmap scan represents the vulnerability and network mapping phase. The FTP brute-force attempt, OS exploit, and SQL injection represent the gaining access phase. The DoS attack and mail blitz attack fulfill the damage-causing motivations of attackers. Finally, the Netcat connection represents the maintaining access phase of the methodology.

4.4 Experiment Design

The overall experiment will require twenty-five tests. Each test will contain ten sessions that may have between one and five random attacks. The attacks during each session will occur at random and may overlap one another. There will be a one minute break between sessions, allowing the attacks during the previous session to resolve before continuing on to the next session. Each session is estimated to run for about twenty-five minutes.

The maximum number of attacks that will be initiated would amount to fifty. Though, the DoS attack may randomly spawn more attacks, raising this number. On average it is expected that each test have at least thirty attacks.

During the tests, Snort will be sniffing traffic, Nmap will be scanning the network, and the modeling software will be correlating the data observed and produce situational awareness models.

Log data will be saved from the Black hat machine specifying which attacks were performed and when they occurred. Model summaries will be saved from the modeling

software indicating what attacks are currently observed. In addition, Wireshark logs of all the network traffic will be saved to validate the attacker logs, assisting in creating the ground truth. The ground truth is set of events that were confirmed to have occurred.

After the experiments are completed, the modeler logs will be compared to the ground truth in order to provide the data for statistical analysis. The next section will explain the performance metrics that will be used during analysis.

4.5 System Boundaries

This section covers the performance metrics that will used to measure the data from these experiments. Also, the system's parameters and factors will be explained. Then, the methods used to apply the metrics will be discussed.

4.5.1 Performance Metrics.

In his research, Dr. George Tadda [54] explains the metrics needed to measure a system that uses data fusion to provide cyber situational awareness. Dr. Tadda addresses four key categories that evaluate the performance and effectiveness such systems: data-to-information ratio, confidence, accuracy, relevance of information, and timeliness. For the timeliness metric, Dr. Tadda explains that it must measure from when an attack is observed to when a response to the attack is performed [54]. This extends beyond the scope of this research, as this research only accounts for observation and orientation of cyber situational awareness. As such, this experiment will only evaluate the system with the first three identified metrics.

4.5.1.1 Data-to-Information Ratio (DIR).

The data-to-information ratio determines how much data was reduced by the modeler from the raw data [54]. Specifically, this metric will measure the amount of event tracks versus the amount of alerts found by the IDS. This metric will show the percentage of data reduction from the raw data.

$$\text{DIR} = \frac{total\ tracks\ observed}{total\ alerts\ observed}$$

4.5.1.2 System Confidence.

System confidence is divided into four metrics by Dr. Tadda [54]. The first metric is recall. Recall is defined as the percentage of the correctly identified observations to the total known number of attacks in the ground truth. This can also be seen as the true positive rate.

$$recall = \frac{total\ tracks\ observed}{total\ tracks\ in\ groundtruth}$$

The next metric that supports system confidence is precision. Precision is the percentage of the correctly identified observations to the total number of observations. Precision is different from recall in that it measures the correctness of observations versus the amount observed.

$$precision = \frac{correct\ tracks}{total\ tracks\ observed}$$

Track fragmentation is the next metric that supports system confidence. As the IDS identifies alerts in the network traffic, the modeling system aggregates alerts into tracks, reducing the amount of redundant information in the situational awareness models. Sometimes, the modeling system may fail to accurately correlated alerts together, creating additional and redundant tracks. The fragmentation rate metric determines the percentage of event tracks that are reported as separate tracks, but should be included in another track. These fragments are identified from comparing the observations against the ground truth.

$$fragmentation = \frac{number\ of\ fragments}{total\ tracks\ observed}$$

The final metric that supports system confidence is the measurement of misassociation. Misassociation is the percentage of incorrectly observed event tracks to the total number of observed tracks. For this system, misassociation determines the correctness of whether the correct signature is applied to each track.

$$misassociation = \frac{tracks\ with\ incorrect\ signatures}{total\ tracks\ observed}$$

4.5.1.3 System Accuracy.

Dr. Tadda [54] separates accuracy, also known as system purity, into two metrics: misassignment rate and evidence recall. The misassignment rate aims to determine the percentage of observed alerts that were erroneously assigned to a specific event track. The erroneous track may be due to fragmentation or misassociation. This shows how accurate the observations are what they are supposed to be.

$$misassignment\ rate = \frac{number\ of\ alerts\ in\ wrong\ tracks}{total\ tracks\ observed}$$

The second metric in system purity, evidence recall, adds to the accuracy of observation identity. Evidence recall is the percentage of the correctly assigned observations to the correct tracks against the total number of observations in the ground truth. This metric details how many of the observed tracks were correctly identified.

$$evidence\ recall = \frac{number\ of\ alerts\ in\ correct\ track}{total\ tracks\ in\ ground\ truth}$$

Tadda and Salerno explained that no value to the SUT's worthiness could be determined, from these two system accuracy metrics. Regardless, it is necessary to include this for future analysis.

4.5.1.4 Relevance of Information.

Dr. Salerno [41] explains that relevance metrics can be used to determine how well the system portrays the criticality of observations. In this research, these metrics will measure specific Activities of Interest (AOI). Dr. Salerno explains that if someone is concerned with only specific activities, this equation will determine the percentage of information that is important. Conversely, it also shows the percentage of unimportant information that someone must acknowledge before finding all of the relevant information. The AOI score is computed using the following equation:

$$\text{AOI } score = \frac{NAOI * NA - \sum_{i=1}^{NAOIR} P_i}{NAOI * NA - \sum_{i=1}^{NAOIR} i}$$

NAOI = number of AOIs in GT

NAOIR = number of AOIs observed

NA = number of activities

P_i = position of i^{th} AOI

4.5.2 Parameters.

The following parameters must be taken into account for the experiment. These parameters will affect the output of the tests.

CPU speed of system server The processor speed is based on the physical hardware of the server. This will not change during the experiments.

CPU utilization of system server The processor of the server is shared between all the guest virtual machines on the virtual network. With more active guests, there is an increase in the wait for clock cycles. During the experiment, no new guests will be added. Though, if one guest has a drastic increase in workload, the other guests may be affected.

CPU utilization of guest machines The processor utilization of an individual virtual machine is based on its workload.

Memory size of system server The server's memory size is based on the physical hardware of the server. This will not change during the experiments.

Memory usage of guest machines The memory utilization of an individual virtual machine is based on its workload.

Throughput of virtual network The network's throughput is the rate that packets can traverse over the network. It can be affected by the frequency of transmissions and the size of packets [62].

Version of guest operating systems The operating systems in the guest machines manage the applications and services. The operating system is an attack vector in cyber attacks. Attacks may or may not succeed depending on the version of the operating system.

Versions of guest applications Like the operating systems, system applications or services are attack vectors. Attacks may or may not succeed depending on the version of the application or service.

IP addresses of guest machines The Internet Protocol (IP) addresses of the hosts affect whether or not they properly receive network traffic. If a packet is intended for a specific network host or a range of hosts, but the IP addresses are incorrect, the host will not receive the packet. In the tests, there will be many packets that never reach their destinations because the destination IP address in the packet does not match an actual host.

Size of model As the system operates, the sensors will observe data and add it to the model. The size of the model affects how long it has to be analyzed, the number of database connections that must be performed, and the time it takes to post it to and download it from the web server.

4.5.3 Factors.

The following factors are a subset of the previous parameters list. The value of these factors will vary across the experiments.

CPU usage of guest machines During the course of the experiments the amount of workload will fluctuate. Scans and file processing will cause the processor utilization

to spike in some machines. This may affect the speed that data is observed on the guest machines.

Memory usage of guest machines Like CPU utilization, memory usage also depends on the host's workload. The virtual machines will each handle their memory and clear it on its own. This again may affect the speed that data is observed on the guest machines.

Throughput of virtual network As the amount of traffic builds from certain attacks, the black hat VM may exhaust its available throughput. This heightened traffic density may also affect the modeling system by slowing its ability to read all of the packets as it sees them; alerts may be delayed during excessive workloads by the black hat VM.

Version of guest operating systems The operating systems of the guest machines are modifiable within the network. Attacks may hook the operating system, resulting in unexpected performance, or the administrator may upgrade the operating system to remove the vulnerability.

Versions of guest applications Applications and services will be attacked during the experiment. During the course of the experiments, the services should become patched and no longer vulnerable against attacks.

Size of model During the experiments, more and more data will be collected. As the amount of data in the model increases, so does the time it takes to analyze the model. The system may have to query the database more to find the answers it needs.

4.5.4 Evaluation Techniques.

To evaluate the SUT, a virtualized enterprise network will act as the environment for the , as previously described. The benefit of using a virtual network is that it is a

real network, versus a simulated one. Being virtualized, the network is controlled and sandboxed. Snapshots of the network can be made and restored for repeated tests.

The metrics for system confidence, accuracy, and the AOI scores will be manually determined. The modeler logs will be analyzed and compared against the ground truth. True positive, false positives, and false negatives will be identified and used in the metrics.

4.6 Limitations

A successful implementation of this system would benefit government and civilian agencies. Though, a direct copy of this system would not be sufficient. This system is designed only as a proof-of-concept system. There are several aspects of this system that does not meet the standards of a fielded product:

- The system was not designed to be secure, which means it most likely has vulnerabilities of its own that can be exploited by attackers.
- The system does not have an assessment or response for every attack. Only responses to predetermined attacks were considered for this experiment. For a fielded system, it must be able to account for a large spectrum of existing exploits as well as be able to update to account for future attacks.
- Speed is a major factor in incident response. Measuring the time from the start of an attack to the start of mitigation procedures can provide insight on how well a system responds. Though, timeliness is not a metric in this experiment.
- The installation of Nagios into the environment proved excessively troublesome. Alternatively, a Java program was built to act in place of Nagios. Even though the data from this program was not tested in this experiment, optimally, Nagios should be added to the system.

4.7 Summary

Achieving cyber situational awareness is a growing concern in government and commercial entities, as is the methods to measure systems that attempt to provide such information. The aforementioned methodology described how to test a system that observes network traffic, consolidates the traffic into specific tracks, and then provides an impact assessment of the tracks. This methodology explained how the SUT would be installed in the test environment, the attacks that would be performed and observed, and finally, how the SUT would be measured.

Success of this system primarily depends on the confidence metrics. Optimally, the true positive rate should far surpass the false positive rate; additionally, the rate should be closer to '1.0' versus being near '0.5', which is equivalent to randomness. Additionally, the DIR and relevance metrics provide insight to how much information or work is saved. For these latter metrics, lower results for are better.

V. Results and Analysis

5.1 Overview

This chapter presents the findings of the tests outlined in chapter four. First, this chapter will provide an overview of the experiment and summarize the data in the ground truth. Next, this chapter will present the results of the metrics and provide the statistical analysis on the data. Finally, this chapter will explain what the data infers about the System Under Test (SUT), which is the cyber situational awareness modeling system detailed in chapter two.

5.2 Implementation

During the experiment, twenty-five tests were run. During each tests, a random number of attacks were performed. The chart on the next page shows the frequency of attacks throughout the twenty-five scenarios making up the ground truth.

In the graph, it is apparent that the attacks do not completely represent randomness. The number of mail blitz attacks appears double because there was an equal chance of an internal mail blitz and an external mail blitz attack, which had to pass through the firewall; this doubled the frequency of mail blitz attacks. Denial-of-Service (DoS) attacks appeared high because with each attack there was a random number of occurrence of simultaneous DoS attempts, creating Distributed Denial-of-Service (DDoS) attacks. In addition, the uncertainty of Metasploit successes resulted in failed exploits using NETBIOS Server Message Block (SMB) buffer overflows, decreasing the number of occurrence for that attack. This can be compared to real world attacks, which also would not occur at an equal frequency.

Overall, 699 of attacks occurred during an aggregated 625-minute timeframe. Over the course of the experiment, there were 113 File Transfer Protocol (FTP) brute-force

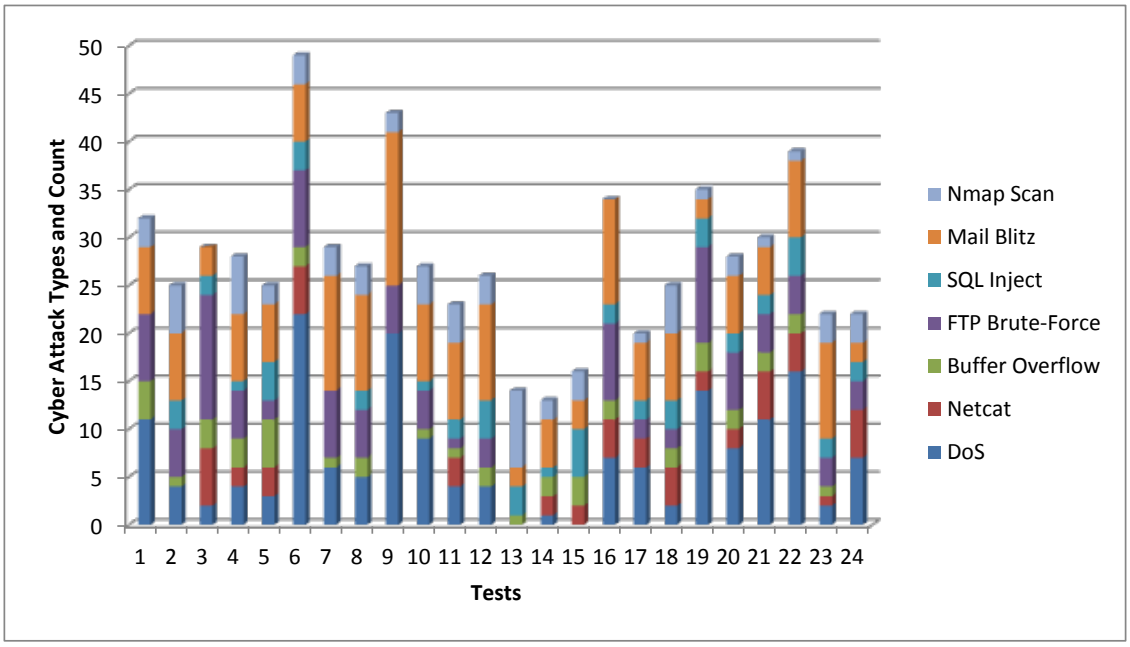

Figure 5.1: Attack frequency in ground truth

attempts, 176 mail blitzes, 183 DoS attempts, 46 NETBIOS SMB buffer overflows, 70 Nmap scans, 55 Structured Query Language (SQL) injections, and 56 Netcat connections. Through analysis of the modeler output against the ground truth, the following tables show the obtained values.

Table 5.1: Raw Data for Samples 1 - 12

Test	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Ground truth count	34	28	30	29	26	50	29	26	43	29	24	27
Total observed	42	38	35	36	33	61	37	34	48	37	31	26
Correct observed	34	28	29	29	25	50	29	24	42	29	24	26
Unidentified	0	0	0	0	0	0	0	1	0	0	0	1
Misidentified	0	1	1	0	1	0	1	1	2	0	0	0
Fragments	7	8	4	4	5	8	5	5	4	5	2	0
Correct alerts	114	100	108	98	60	120	127	99	136	120	84	102
Incorrect alerts	23	25	13	24	25	25	17	16	15	21	8	0

Table 5.2: Raw Data for Samples 13 - 25

Test	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
Ground truth count	14	13	16	34	20	24	36	29	31	38	22	22	37
Total observed	16	14	17	42	21	31	40	35	35	47	29	21	42
Correct observed	14	13	16	34	20	24	35	29	30	37	21	21	35
Unidentified	0	0	0	0	0	0	1	0	1	0	0	1	1
Misidentified	0	0	0	1	0	1	0	0	0	1	1	0	2
Fragments	2	1	1	6	1	4	3	2	4	4	4	0	2
Correct alerts	34	42	33	100	49	75	77	107	75	81	82	68	100
Incorrect alerts	4	2	2	17	4	20	10	6	8	19	22	0	19

The values from the above tables were applied to the metrics outlined in section 4.5. The results of these metrics are detailed in the following sections.

5.3 Data Reduction Results

This section will present the results for the aforementioned metrics. As the data mostly appears in ratios, pie graphs best describe the data. In addition, normality will be tested using the Shapiro-Wilk test. For this test, the null hypothesis aims for the data to come from a population that is normally distributed. Rejection of the null hypothesis can be attributed to the W value being small or if the P-value is less than the alpha value. The alpha value for these tests will be 0.05. The distributions for each of the metrics will be shown in a normal quantile plot with bar graphs.

5.3.1 Data-to-Information Ratio (DIR).

For the malicious traffic identified by Snort, the modeling program reduced the data presented to the user down to 34.55% of the original amount. This is portrayed in the figure below.

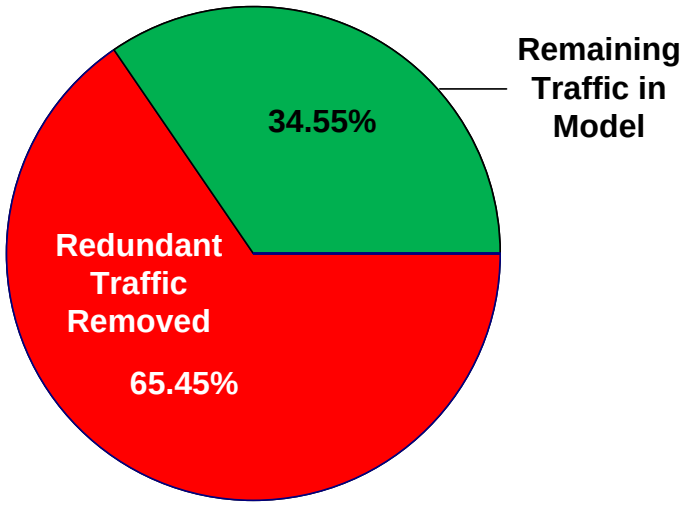

Figure 5.2: Data-to-Information Ratio

This reduction of information saves the end user or their systems 65.45% of the time and resources they would take processing the data. The following distribution shows that the data resembles normality.

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.917645 and the P value was 0.0453. Since the P value rounds up to the alpha level, then we fail reject the null hypothesis, which tells us that the distribution is just barely normal using the twenty-five samples.

With 95% confidence, the upper confidence interval is 37.39% and the lower confidence interval is 31.71%.

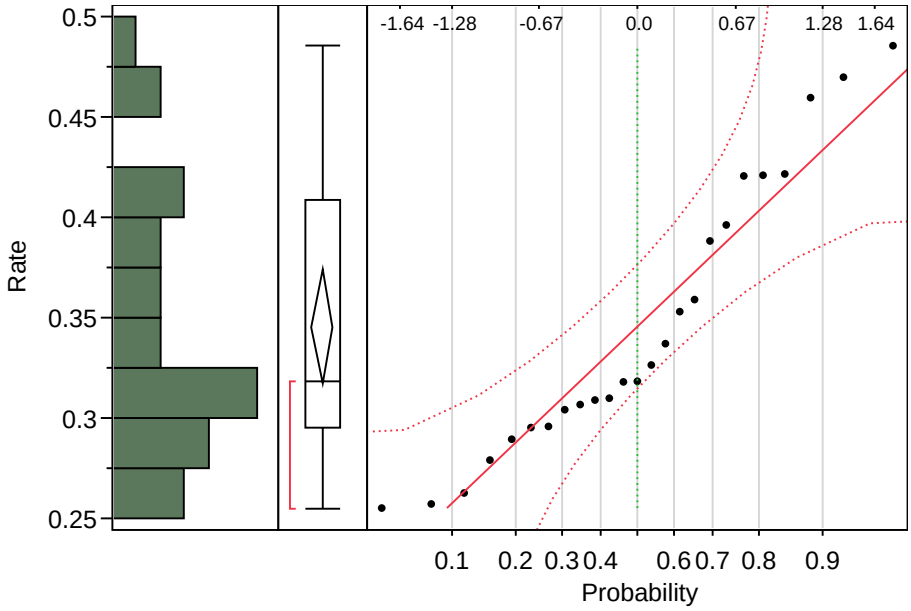

Figure 5.3: Distribution for Data-to-Information Ratio

This saves time wasted and allows the user to allocate their resources to more productive ventures.

5.4 Confidence Results

To determine the confidence of the system, the metrics for recall, precision, fragmentation rate, and misassociation are computed. These metrics depend on the values for the total number of tracks in the ground truth, the total number of tracks observed, the total number of correctly identified tracks, the total number of track fragments, and the total number of misidentified tracks.

5.4.1 Recall.

Recall has a mean of 98.24% with a standard deviation of 2.26% as shown in the pie chart below. This shows that 98.24% if the attacks in the ground truth were accounted for by the system. The following plot shows that the distribution does not represent normality.

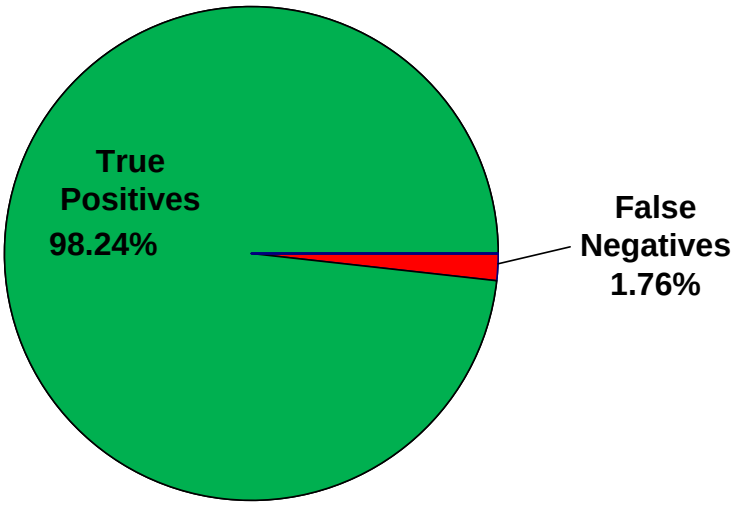

Figure 5.4: System Recall Rate

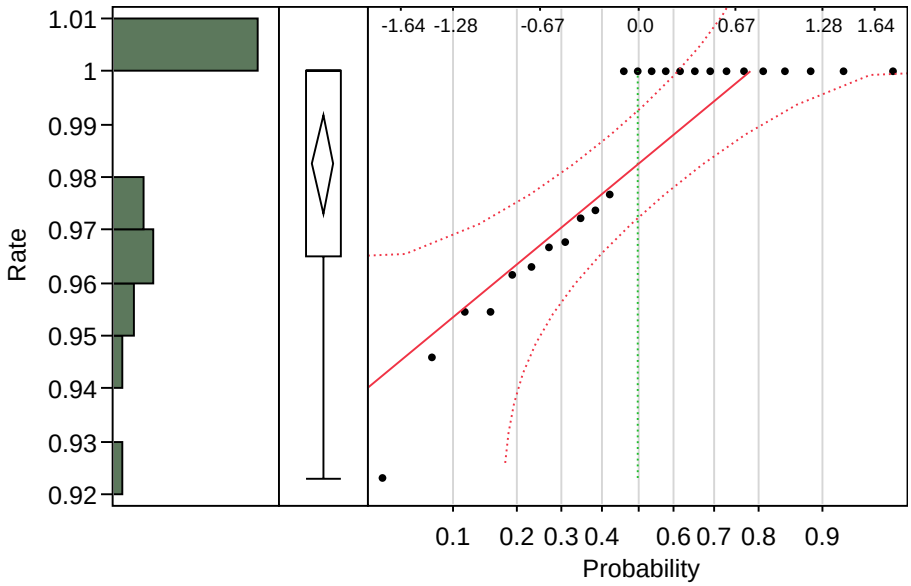

Figure 5.5: Distribution for Recall Values

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.775605 and the P value

was 0.001. Since the P value is less than the alpha level, we reject the null hypothesis, which tells us that the distribution does not yet appear normal. The high number of perfect samples is the cause of the normality failure. Since many of the samples contained observations in the upper twenties or thirties, a single error results in a recall rate in the mid-nineties. This leaves a gap in the upper nineties. Since perfect recall is desired, normality should not be seen with this number of observations in the ground truth. If the number of observations reached near one hundred for each sample, we would see normality.

With 95% confidence, the upper confidence interval is 99.17% and the lower confidence interval is 97.31%.

5.4.2 Precision.

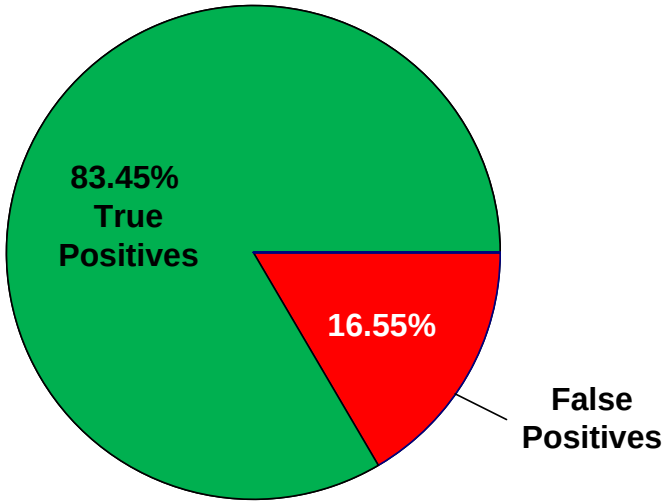

Figure 5.6: System Precision Rate

Precision has a mean of 83.45% with a standard deviation of 8.06%. This tells us that 83.45% of the observations were actual attacks in the ground truth. The false positives

can mostly be attributed to the fragmented tracks, detailed in the next measurement. The following plot shows that the distribution is normal.

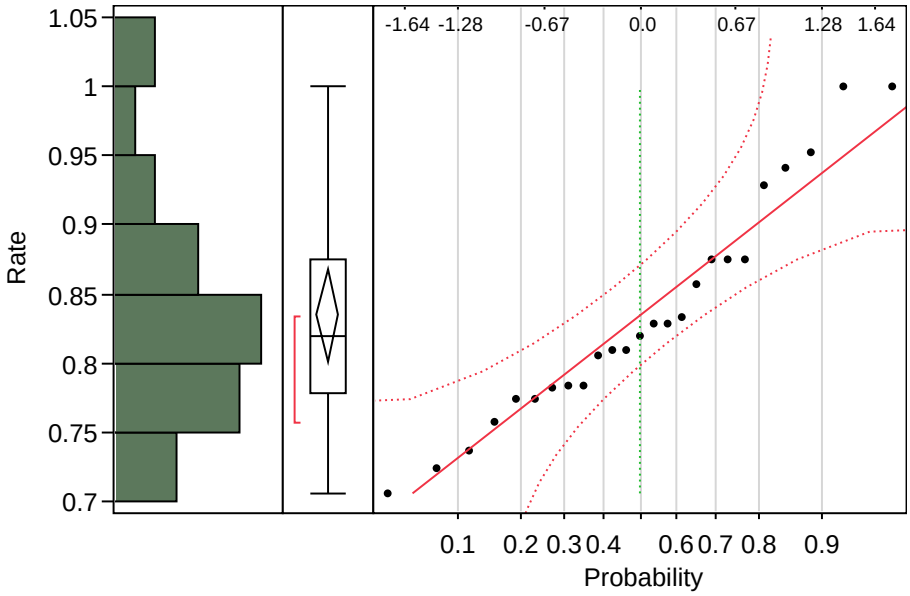

Figure 5.7: Distribution for Precision Values

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.945630 and the P value was 0.1995. Since the P value is greater than the alpha level, then we fail reject the null hypothesis, which tells us that the distribution represents normality.

With 95% confidence, the upper confidence interval is 86.76% and the lower confidence interval is 80.12%.

5.4.3 Fragmentation Rate.

The fragmentation rate has a mean of 10.17% with a standard deviation of 5.10%, displayed in the following pie chart.

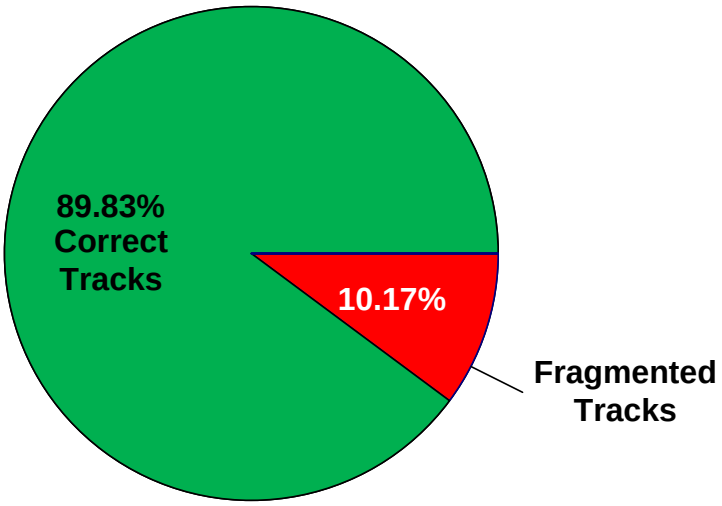

Figure 5.8: System Fragmentation Rate

This measurement means that 10.17% of all correctly identified attacks fragmented into additional alerts, adding to the false positive rate.

The following plot shows that the distribution for the fragmentation rate is normal. Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.964792 and the P value was 0.5179. Since the P value is greater than the alpha level, then we fail reject the null hypothesis, which tells us that the distribution shows normality.

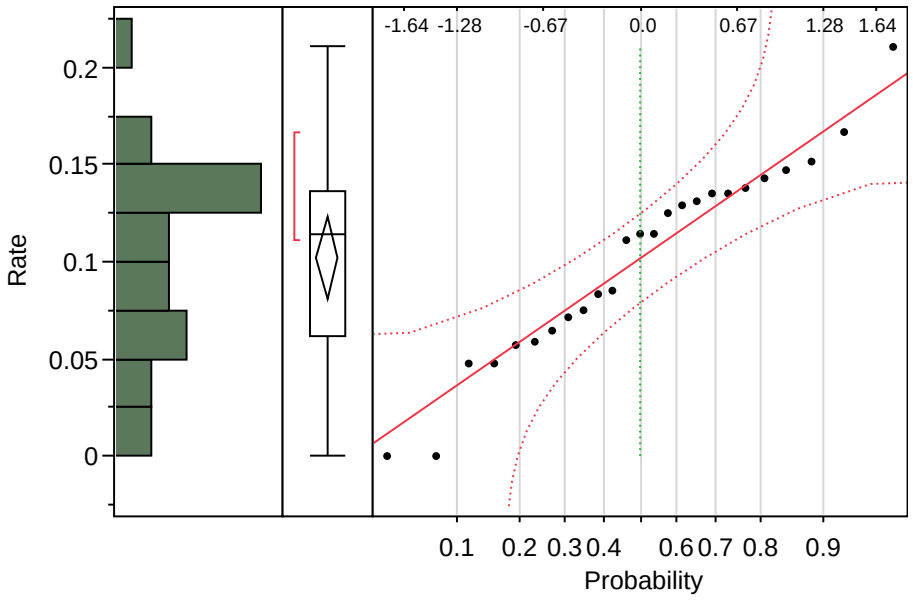

Figure 5.9: Distribution for fragmentation values

With 95% confidence, the upper confidence interval is 12.27% and the lower confidence interval is 8.06%.

5.4.4 Misassociation Rate.

The misassociation rate has a mean of 1.29% with a standard deviation of 1.56%. This means that 1.29% of all observations by the system was incorrectly identified as a different type of attack. For instance, a mail blitz attack may have been identified as a DoS attempt. This rate is shown by the pie chart on the next page.

The quantile plot on the next page shows that the distribution is not normal. Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.760982 and the P value was 0.0001. Since the P value is less than the alpha level, we reject the null hypothesis, which tells us that the distribution is not normal. As with the other normality test for the recall rate, the failure of normality is the result on too many perfect samples.

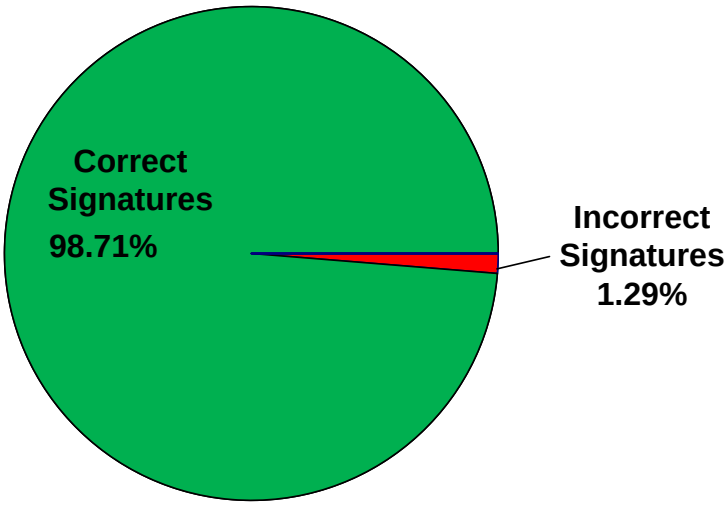

Figure 5.10: System Misassociation Rate

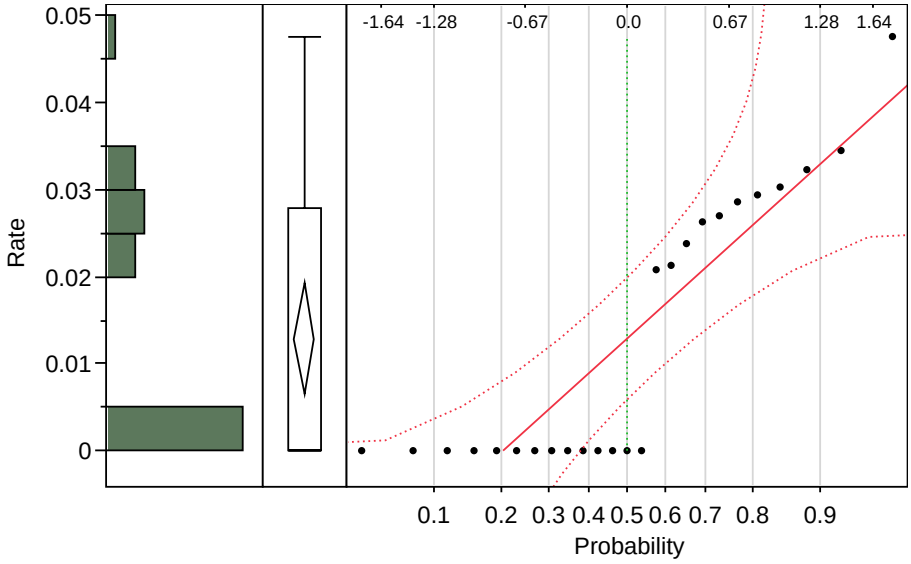

Figure 5.11: Distribution for missassociation values

With 95% confidence, the upper confidence interval is 1.93% and the lower confidence interval is 0.64%.

5.4.5 True Positive Rate versus False Positive Rate.

To determine the overall accuracy of the system, a Receiver Operator Characteristic (ROC) curve is used to plot the recall rate versus the precision rate. The 45 degree angle line is the threshold representing randomness. The closer the curve is to the line, the least accurate the system is. If the curve falls below the line, it indicates the system performs worse than random. The closer the curve is to the upper-left corner, the more accurate the system is. The accuracy of this system is plotted on the following curve.

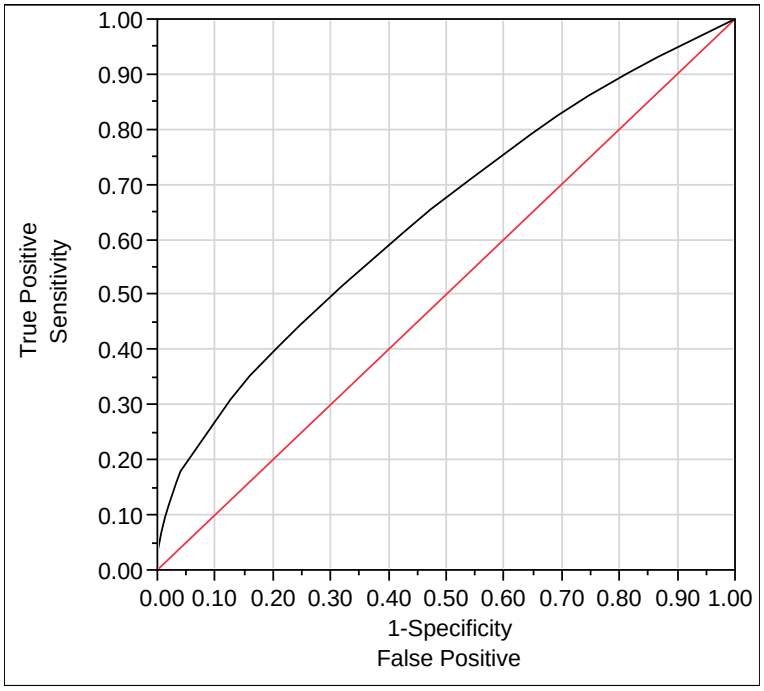

Figure 5.12: ROC Curve for System Accuracy

As seen in the diagram, the ROC curve is above the threshold, showing the system performs better than random. The area under the curve is at 64.07% of the total area. This shows that the modeling system is 64.07% accurate, which is better than randomness. The false positive rate of 35.93% is mostly attributed to the high fragmentation rate.

5.5 System Purity Results

The next set of metrics addresses the purity of event tracks. These metrics include the misassignment rate of alerts and the percentage of correctly assigned alerts. These metrics depend on the values for the total number of tracks observed, the total number of alerts identified in an incorrect track signature, and the total number of identified alerts under the correct track.

5.5.1 Misassignment Rate.

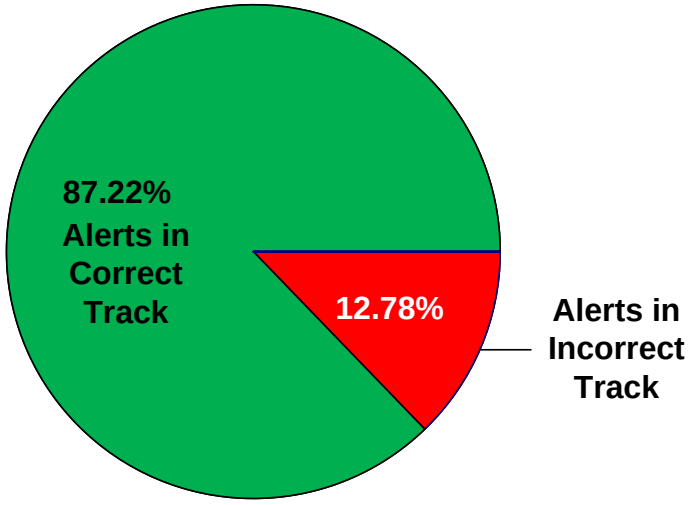

Figure 5.13: System Misassignment Rate

The misassignment rate has a mean of 12.78% with a standard deviation of 7.02%. This shows that 12.78% of all observations from the Intrusion Detection System (IDS) were not assigned to the correct track, but instead to a fragmented track or to a misidentified track. The following plot shows that the distribution is normal.

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.980904 and the P

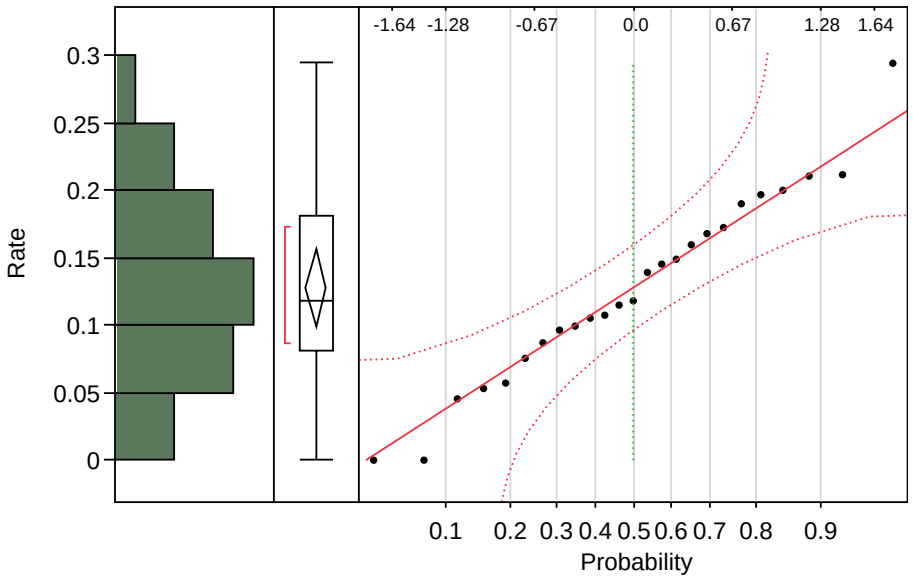

Figure 5.14: Distribution for Misassignment Values

value was 0.9024. Since the P value is greater than the alpha level, we fail to reject the null hypothesis, which tells us that the distribution represents normality.

With 95% confidence, the upper confidence interval is 15.68% and the lower confidence interval is 9.89%.

5.5.2 Evidence Recall.

The evidence recall rate has a mean of 86.98% with a standard deviation of 6.84%. This means that the system correctly portrayed 86.98% of the attacks from the ground truth. The remaining 13.02% is again mostly attributed to the fragmentation rate. The following plot shows that the distribution is normal.

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.979563 and the P value was 0.8763. Since the P value is greater than the alpha level, and then we fail to reject the null hypothesis, which tells us that the distribution is normal.

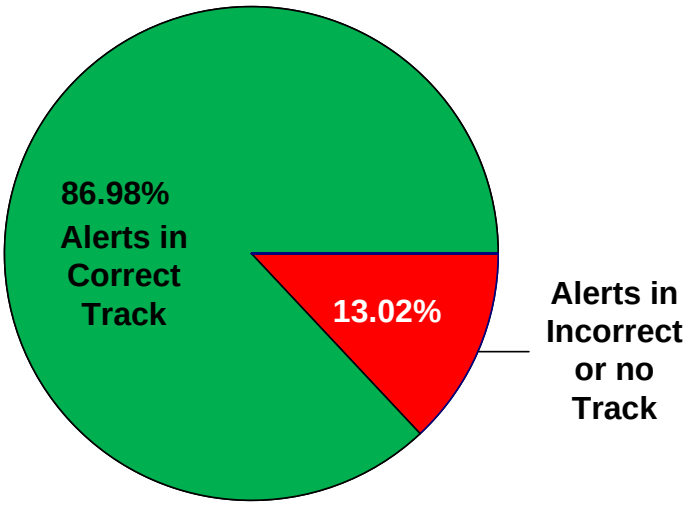

Figure 5.15: System Evidence Recall Rate

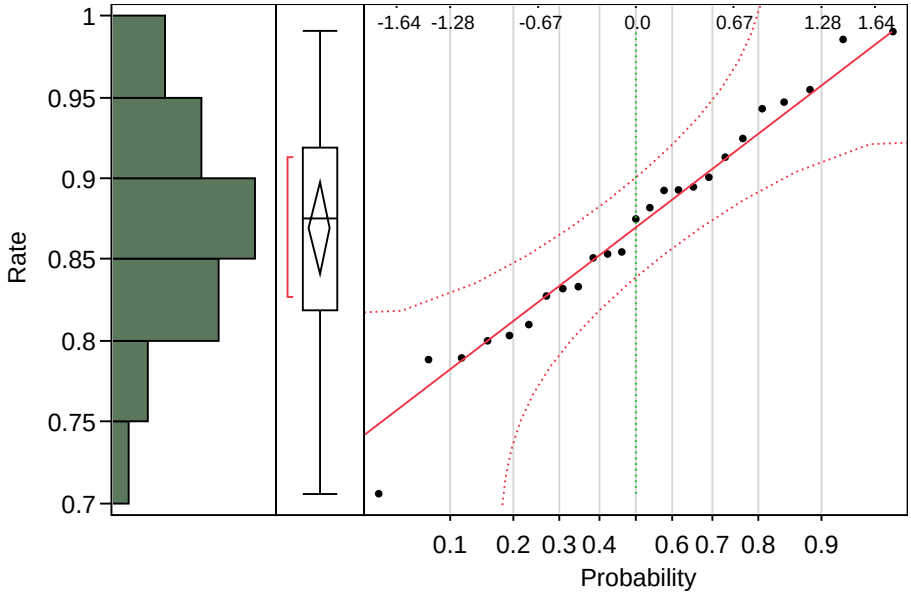

Figure 5.16: Distribution for Evidence Recall Values

With 95% confidence, the upper confidence interval is 89.80% and the lower confidence interval is 84.16%.

5.6 Information Relevance Results

5.6.1 High Impact Activities of Interest.

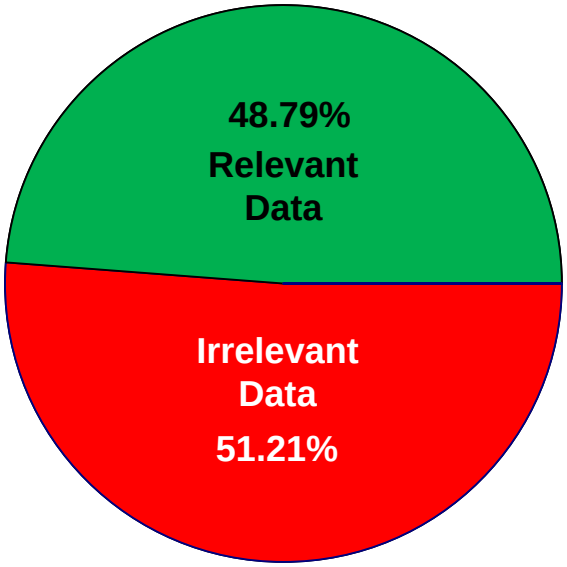

Figure 5.17: High Impact AOI Relevancy Ratio

The activity of interests scores for high impacts have a mean of 48.79% with a standard deviation of 17.44%. For the cyber situational awareness of this system, only 48.79% of the information is important when concerned with high impact activities; 51.21% of the information is not of interest. This shows that when the network responder does not know the impact, then they would have to acknowledge or resolve 51.21% of the irrelevant activities in order to respond to all of the high ranking events. As this system presents the impact to the responder when the event appears, it shows that this system reduces the responder's workload by 51.21% when they are only concerned with high impact events. The following plot shows that the distribution resembles normality.

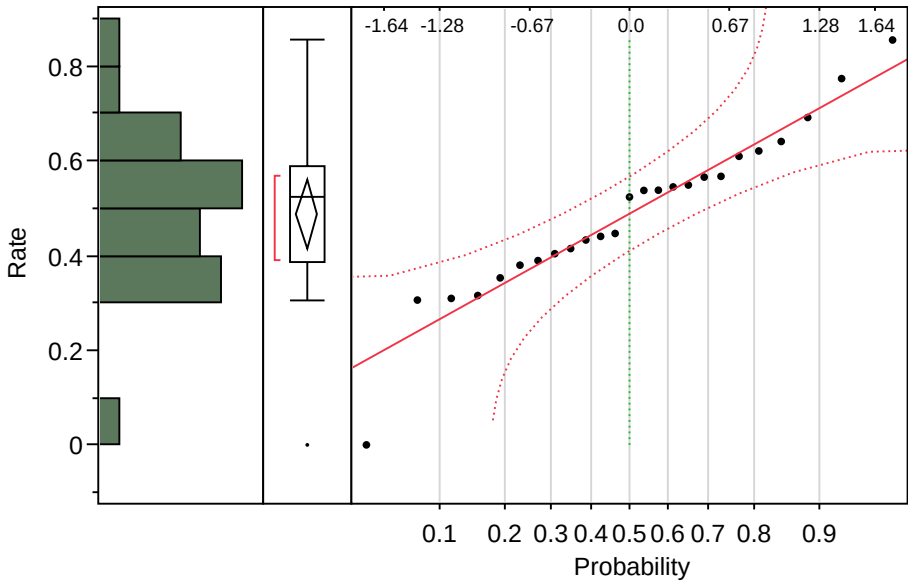

Figure 5.18: Distribution for High Impact AOI Values

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.961039 and the P value was 0.4356. Since the P value is greater than the alpha level, we fail to reject the null hypothesis, which tells us that the distribution is normal.

With 95% confidence, the upper confidence interval is 55.99% and the lower confidence interval is 41.59%.

5.6.2 High-Medium Impact Activities of Interest.

The activity of interests scores for high and medium impacts have a mean of 65.97% with a standard deviation of 16.19%. For the cyber situational awareness of this system, only 65.97% of the information is important when concerned with high and impact activities; 34.03% of the information is not of interest, or only concerning low impact activities. This shows that when the network responder does not know the impact, then they would have to acknowledge or resolve 34.03% of the irrelevant activities in order to respond to all of the high and medium ranking events. As explained in the previous section,

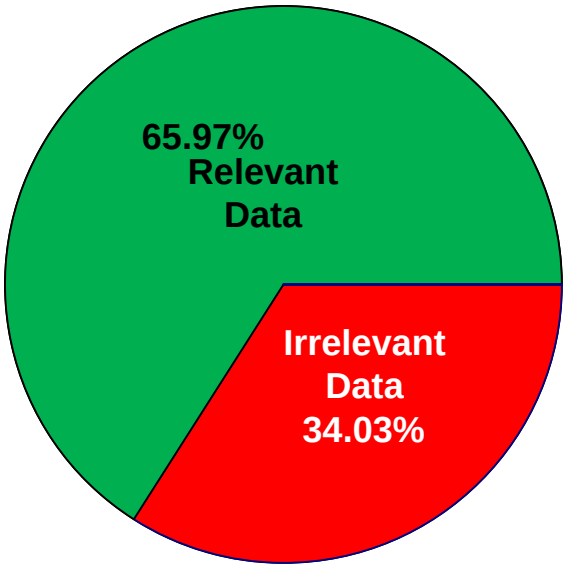

Figure 5.19: High-Medium Impact AOI Relevancy Ratio

it shows that this system reduces the responder's workload by 34.03% when they are only concerned with high impact events. The plot on the next page shows that the distribution resembles normality.

Using the Shapiro-Wilk test with an alpha level of 0.05, I tested the null hypothesis to determine if the distribution is normal. For this data, W equaled 0.985018 and the P value was 0.9634. Since the P value is greater than the alpha level, we fail to reject the null hypothesis, which tells us that the distribution is normal.

With 95% confidence, the upper confidence interval is 72.65% and the lower confidence interval is 59.28%.

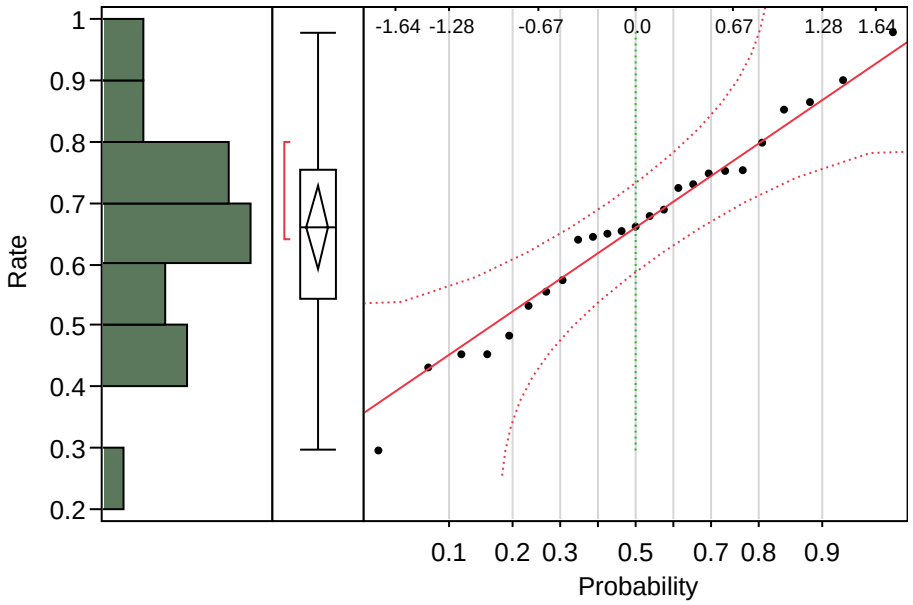

Figure 5.20: Distribution for High-Medium Impact AOI Values

5.7 Summary

The results in this chapter show the performance and the usefulness of using the cyber situational awareness modeling system. Of significance is the overall accuracy of the system, the amount of data reduction, and the relevance of using the impact assessment system.

Due to alert correlation, the amount of alerts is reduced by 65.45%. After reduction, only 34.55% of the original data is presented to the end user. This saves time and resources from having to parse though the redundant alerts and determining if the alerts are related.

For the true positive versus false positive rate, the overall accuracy of the system is 64.07%. This shows that using this system is advantageous over not using a situational awareness modeling system.

By using the impact assessment system, the modeler tells the user if the attacks are high, medium, or low impacts to the network. If the user is only interested in specific levels, they can ignore the other attacks, thus saving significant resources. If the user is

only interested in activities that are high impact, this experiment shows that only 48.79% of the data is significant and 51.21% is irrelevant. If they are concerned with high and medium impacts, 65.97% of the information is relevant, while 34.03% can be ignored. Without having to parse through the irrelevant data, the user can save significant time and resources administering the network.

These results show that this system does have a significant impact versus not using the system at all. Since the metrics used to measure this system are newly developed by Air Force Research Laboratories (AFRL) and have not been used before, there is no data to compare these results. Though, future systems that use these metrics can be compared to this system.

VI. Conclusion

6.1 Overview

This chapter will reiterate the results of the experiment and how the results quantify and qualify the proposed and constructed network situational awareness modeling system. Also explained is how this research contributes to the information technology (IT) community, to the Air Force, and to Air Force Institute of Technology (AFIT). In addition, this chapter contains details on how this system should be used in future research, what is missing in the system, and how the modeling system can be improved. Finally, the last section proposes research topics that extends and can be built upon this system.

6.2 Test Results

The results in this research show the performance and the usefulness of using the cyber situational awareness modeling system developed in chapter three. Of significance is the overall accuracy of the system, the amount of data reduction, and the relevance of using the impact assessment system.

The tests lend great credibility to the impact awareness system. From the alert correlation process, the amount of alerts was reduced by 65.45%. That means the system users only had to acknowledge 34.55% of the original data that was observed by the Intrusion Detection System (IDS). This process would save time and resources in operational systems.

By using the impact assessment system, the modeler informs the users if the attacks are high, medium, or low impact, and if their systems are even vulnerable. If the user is interested in specific impact, they can ignore the other IDS observations. In the experiment, when the user was only interested in high impact attacks, the results showed that only 48.79% of the data were relevant while 51.21% was irrelevant. When the users were

concerned with both high and medium impacts, 65.97% of the information was significant, while 34.03% was not important. By using the impact assessment system, users could ignore irrelevant observations and save significant time.

These results show that this system is an effective cyber situational awareness modeling tool. These results establish a baseline for comparison with future systems.

6.3 Contributions

This research contributes to the United States Air Force and to the IT community in several ways. Firstly, a process was developed that can guide researchers to achieve basic cyber situational awareness in their networks. A situational awareness modeling tool was developed that provided correlated information about a network. Air Force researchers will have this tool available, enabling them to accelerate their research.

Furthermore, the system provided real-time impact assessments of cyber attacks with regards to the actual computers in the network. The impact scoring system is unique and can be implemented in other cyber networks. This impact assessment process reduced the time for defenders to respond to attacks by showing them what they should be concerned with and what information is irrelevant.

In the end, this research sets a foundation for research and development in using real networks and tools to pursue development and advancements in cyber attack and defense.

6.4 System Limitations

The section describes the aspects of the system that did not perform as planned or could have been improved.

System should utilize other different sensors. Using only Nmap with PBNJ to inventory the network and services provided limited information. For the purpose of this system, the vulnerability assessment on the attacks performed as predicted. Though, when guarding against Operating System (OS) exploits, to include malware, a more in-

depth vulnerability assessment tool would be necessary. It would be recommended that Nmap be replaced or complemented with the Nexpose or Open Vulnerability Assessment System (OpenVAS) tools to provide detailed system security holes and network mapping to modeling system.

AVG antivirus experienced issues during the experiment. It refused to respond after a period of use and had to be restarted. This problem occurred on all virtual machines using AVG and became predictable. It is recommended that this issue be solved or an alternative to AVG be used.

A host monitoring program like Nagios should be used instead of the custom-built monitoring program implemented in the experiment.

Upgrading these three aspects of the system would allow this tool to provide the most accurate information about a cyber network.

6.5 Future Work

Upon researching data fusion, correlation, and information presentation, as well as developing the network modeler in this research, there were several research topics and developmental vectors that were desired but not followed due to time constraints. This section describes these topics.

This research only accounted for level zero, level one, and level three of the five levels in the Joint Director's of Laboratories (JDL) data fusion model [64]. Level two, situation assessment, and level four, future state predictions, should be incorporated to complete the framework.

Event track correlation is scripted for this system for specific IDS observations. For persistent attacks, a general correlation method is used, but for multistage attacks with different signatures, only Nmap scans, FTP Brute-Force attempts, and Mail Blitz events for this specific environment were designed to fuse. More advanced event track correlation techniques are needed. Also, the correlation process separated attacks into different tracks

after a period of inactivity was observed. A period of fifteen seconds was found that reduced fragmentation the most without merging additional observations, but this does not lend itself to attacker profiling. Instead, all alerts from a specific source should be correlated into a single track, regardless of time. This can potentially provide detailed activities of attackers and help predict their intent. Using a machine learning agent can create a dynamic and robust correlation agent.

In chapter two, vulnerability and bug tracking databases were described. This research used the Common Vulnerabilities and Exposures (CVE) repository and scoring system to rate the severity of attacks used during the experiment. Only a select few attacks were used during the experiment. A more robust system should be able to handle a wider variety of attacks and update in real time. An interface to the online vulnerability repositories would be optimal. An automated, intelligent agent should be able to check for updates online and modify the modeling system accordingly to account for the new attacks or vulnerabilities.

As mentioned in the previous paragraph, only a few attacks were watched for during the experiment. In the Snort intrusion detection system, most of the rules to identify a variety of attacks were disabled. For a usable system, many more of the Snort rules should be used in addition to many of the community rules. Alternate or additional intrusion detection systems may be used to increase the accuracy of identifying attacks. Integration of a fully operable and robust intrusion detection system is necessary to detect events in networks to fulfill the community guidelines of data fusion.

As explained in chapter four, the experiment only measured the IDS alert correlation into tracks and the impact assessment process. The cyber situational awareness modeling tool explained in chapter three had many more components not analyzed. Future work could test the accuracy and usefulness of data acquired from the host monitoring software, antivirus software, and the action identification process.

This research focused entirely on identifying and orienting network traffic and attack data. This is only accounts for the first two phases of the Observe, Orient, Decide, and Act (OODA) loop. The ”decide” and ”act” phases were not addressed. A robust responsive framework must be developed to fulfill this need. A human response may or may not be sufficient to defend against the cyber attacks seen in today's networks. Research is needed to determine whether artificially intelligent agents are superior to human administrators in network defense. In addition automated systems should be analyzed to determine how they can extend this modeling system to fulfill the ”decide” and ”act” phases of the OODA loop.

This system was developed to provide situational awareness in a single Local Area Network (LAN). Much larger networks require such situational awareness such as Department of Defense (DoD), government, and corporate networks. A vital future work would be to determine the how this system would scale up to larger and distributed networks. This system may optimally work on a single LAN, so multiple systems may be required that reports information to a central node.

The recommendations in this section specifically aim to improve this cyber situational awareness tool, but there are many more avenues to explore in cyber situational awareness that can add to this research. As situational awareness in cyberspace is growing in demand, there is no foreseeable limit to the potential of future work.

6.6 Summary

In conclusion, this framework for modeling network situational awareness proved to be a success as seen through the experiment's results. The recall and precision rates could not be compared against any previous work, but the metrics showing data reduction proved that this process and tool will save resources for cyber operators. This research provides a foundation for cyber situation awareness and enables network defense tools to protect networks.

Bibliography

[1] Abraham, J. D. "PBNJ 2.04", 2012. URL http://pbnj.sourceforge.net.

[2] Argauer, B. and S. Yang. "VTAC: virtual terrain assisted impact assessment for cyber attacks". *Security and Defense Symposium, Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security Conference, in proceedings of SPIE*. 2008.

[3] BackTrack Linux. "BackTrack Linux - penetration testing distribution", 2012. URL http://www.backtrack-linux.org/.

[4] Blasch, E., I. Kadar, J. Salerno, M. Kokar, S. Das, G. Powell, D. Corkill, and E. Ruspini. "Issues and challenges of knowledge representation and reasoning methods in situation assessment (Level 2 Fusion)". 623510–623510–14, 2006. URL +http://dx.doi.org/10.1117/12.669779.

[5] Blasch, E., J. Salerno, and G. Tadda. "Measuring the worthiness of situation assessment". *Aerospace and Electronics Conference (NAECON), Proceedings of the 2011 IEEE National*, 87 –94. july 2011. ISSN 0547-3578.

[6] Blasco, J. "emerging-all.rules", 2012. URL http://rules.emergingthreats.net/.

[7] Boyd, J. R. "A discourse on winning and losing", 1987.

[8] Chuvakin, A. and V. Myasnyankin. "Complete Snort-based IDS Architecture, Part Two", 2002.

[9] Cole, E. *Network Security Bible*. Wiley Publishing, Inc., Indianapolis, IN, 2nd edition, 2009.

[10] Endsley, M. R. "Toward a theory of situation awareness in dynamic systems". *Human factors and ergonomics society*, 37(2):32–64, March 1995.

[11] Fava, D. S., S. R. Byers, and S. J. Yang. "Projecting Cyberattacks Through Variable-Length Markov Models". *Information Forensics and Security, IEEE Transactions on*, 3(3):359–369, 2008. ID: 1.

[12] Gao, W., J. Wen, N. Jiang, and H. Zhao. "A Study of Data Fusion Based on Combining Rough Set with BP Neural Network". *Hybrid Intelligent Systems, 2009. HIS '09. Ninth International Conference on*, volume 3, 103–106. 2009. ID: 1.

[13] Gates, R. M. "Submitted statement to the Senate Armed Services Committee". January 2009.

[14] Gupta, V. "The Nagios Setup Explained", 2011. URL http://www.linuxforu.com/2011/07/nagios-setup-guide/.

[15] H., Chengchen, L. Zhen, Z. Chen, and B. Liu. "On the deployment strategy of distributed network security sensors". *Networks, 2005. Jointly held with the 2005 IEEE 7th Malaysia International Conference on Communication., 2005 13th IEEE International Conference on*, volume 1, 6 pp. nov. 2005. ISSN 1531-2216.

[16] Hu, C., Z. Liu, Z. Chen, and B. Liu. "On the deployment strategy of distributed network security sensors". *Networks, 2005. Jointly held with the 2005 IEEE 7th Malaysia International Conference on Communication., 2005 13th IEEE International Conference on*, volume 1, 6 pp. nov. 2005. ISSN 1531-2216.

[17] Jajodia, S. and S. Noel. *Advanced Cyber Attack Modeling, Analysis, and Visualization*. Final AFRL-RI-RS-TR-2010-078, Air Force Research Labs, 2010.

[18] Ji, J. W. *Holistic Network Defense: Fusing Host and Network Features for Attack Classification*. Master's thesis, Air Force Institute of Technology, March 2011.

[19] Khouzani, M. H. R., S. Sarkar, and E. Altman. "A dynamic game solution to malware attack". *INFOCOM, 2011 Proceedings IEEE*, 2138–2146. 2011. ISBN 0743-166X. ID: 1.

[20] Klein, G., J. Tolle, and P. Martini. "From detection to reaction - a holistic approach to cyber defense". *Defense Science Research Conference and Expo (DSR), 2011*, 1–4. 2011. ID: 1.

[21] Larsen, J. and J. Haile. "Understanding IDS Active Response Mechanisms", 2010. URL http://www.symantec.com/connect/articles/understanding-ids-active-response-mechanisms.

[22] Liu, X., H. Wang, J. Lai, Y. Liang, and C. Yang. "Multiclass Support Vector Machines Theory and Its Data Fusion Application in Network Security Situation Awareness". *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, 6349 –6352. sept. 2007.

[23] Lye, K. and J. Wing. "Game Strategies in Network Security", 2002.

[24] Lynn, W. J. "Defending a new domain: the Pentagon's cyberstrategy". *Foreign Affairs*, 89(5):97–108, 2010.

[25] Mathew, S., D. Britt, R. Giomundo, S. Upadhyaya, M. Sudit, and A. Stotz. "Real-time multistage attack awareness through enhanced intrusion alert clustering". *Military Communications Conference, 2005. MILCOM 2005. IEEE*, 1801 –1806 Vol. 3. oct. 2005.

[26] Maybury, M. T. "Air Force Cyber Vision 2025", 2012.

[27] Mell, P., K. Scarfone, and S. Romanosky. "The Common Vulnerability Scoring System (CVSS) and Its Applicability to Federal Agency Systems", 2007.

[28] Milner, R. *Communication and Conurrency*. Prentice Hall International, Great Britian, 1989.

[29] MITRE. "Common Vulnerabilities and Exposures", 2012. URL http://cve.mitre.org/.

[30] Nagios Enterprises. "Nagios.org", 2012. URL http://www.nagios.org/.

[31] Natarajan, R. "Top 5 best system monitoring tools". *The Geek Stuff*, 2009. URL http://www.thegeekstuff.com/2009/09/top-5-best-network-monitoring-tools/.

[32] National Institute of Standards and Technology. "National Vulnerabilities Database", 2012. URL http://nvd.nist.gov/.

[33] Nmap.org. "Nmap.org", 2012. URL http://nmap.org/.

[34] Onwubiko, C. "Functional requirements of situational awareness in computer network security". *Intelligence and Security Informatics, 2009. ISI '09. IEEE International Conference on*, 209 –213. june 2009.

[35] OpenVAS. "Open Vulnerability Assessment System", 2013. URL http://www.openvas.org/.

[36] Pfleeger, C. P. and S. L. Pfleeger. *Security in Computing*. Prentice Hall, Upper Saddle River, New Jersey, 4th edition, 2006.

[37] Phister, P. W. "Cyberspace: the ultimate complex adaptive system". *The International C2 Journal*, 4(2):1–30, 2011.

[38] Rapid7. "Vulnerability Management Software - Nexpose", 2013.

[39] Russell, S. and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Upper Saddle River, New Jersey, 3rd edition, 2010.

[40] SAINT Corporation. "SAINT", 2013. URL http://www.saintcorporation.com/.

[41] Salerno, J. "Measuring situation assessment performance through the activities of interest score". *Information Fusion, 2008 11th International Conference on*, 1 –8. 30 2008-july 3 2008.

[42] Salerno, J., M. Sudit, S. Yang, G. Tadda, I. Kadar, and J. Holsopple. "Issues and challenges in higher level fusion: Threat/impact assessment and intent modeling (a panel summary)". *Information Fusion (FUSION), 2010 13th Conference on*, 1–17. july 2010.

[43] Sawilla, R. E. and D. J. Wiemer. "Automated computer network defence technology demonstration project (ARMOUR TDP): Concept of operations, architecture, and integration framework". *Technologies for Homeland Security (HST), 2011 IEEE International Conference on*, 167–172. 2011. ID: 1.

[44] Schultz, E. "The importance of situational awareness". *Network security consulting blog: articles by network security consultants*, 2010. URL http://http://blog.emagined.com/2010/04/09/the-importance-of-situational-awareness/.

[45] SecTools.Org. "SecTools.Org: Top 125 Network Security Tools", 2013. URL http://sectools.org/.

[46] Security Focus. "Bugtraq", 2012. URL http://www.securityfocus.com/archive/1.

[47] Shiravi, H., A. Shiravi, and A. Ghorbani. "A Survey of Visualization Systems for Network Security". *Visualization and Computer Graphics, IEEE Transactions on*, PP(99):1–1, 2012. ID: 1.

[48] Skoudis, E. and T. Liston. *Counter hack reloaded: a step-by-step guide to computer attacks and effective defenses*. Prentice Hall, Upper Saddle River, New Jersey, 2nd edition, 2009.

[49] Sourcefire. "Snort", 2012. URL http://www.snort.org/.

[50] Steinberg, A., C. Bowman, and F. White. "Revisions to the JDL data fusion model". Belur V. Dasarathy (editor), *Sensor fusion: architectures, algorithms, and applications III. SPIE Proceedings Vol. 3719*, volume 3719, 430–441. 1999.

[51] Symantec. *Symantec Intelligence Quarterly: July - September, 2011*. Whitepaper, Symantec Corporation, 09 2011.

[52] Symantec. "Symantec DeepSight Threat Management System", 2012. URL https://tms.symantec.com/.

[53] Symantec. *Symantec Intelligence Report: July, 2012*. Technical report, Symantec Corporation, 07 2012.

[54] Tadda, G. "Measuring performance of Cyber situation awareness systems". *Information Fusion, 2008 11th International Conference on*, 1–8. 30 2008-july 3 2008.

[55] Tadda, G. and J. Salerno. "Overview of Cyber Situation Awareness". *Cyber Situational Awareness*, 15–35. 2010.

[56] Tadda, G., J. Salerno, D. Boulware, M. Hinman, and S. Gorton. "Realizing situation awareness in a cyber environment". 624204–624204–8, 2006. URL +http://dx.doi.org/10.1117/12.665763.

[57] Tenable Network Security. "Nessus 5.0 User Guide", 2012. URL www.tenable.com.

[58] Thomas, C. and N. Balakrishnan. "A Survey of Visualization Systems for Network Security". *Information Forensics and Security, IEEE Transactions on*, 4(3):542–551, 2009.

[59] Thomas, R. W., L. A. DaSilva, and A. B. MacKenzie. "Cognitive networks". *New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005. 2005 First IEEE International Symposium on*, 352–360. 2005. ID: 1.

[60] Tidwell, T., R. Larson, K. Fitchand, and J. Hale. "Modeling Internet Attacks". *Workshop on Information Assurance and Security, Proceedings of the 2001 IEEE*, 54–59. 2001.

[61] Tran, K. and H. Jin. "Detecting Network Anomalies in Mixed-Attribute Data Sets". *Knowledge Discovery and Data Mining, 2010. WKDD '10. Third International Conference on*, 383–386. 2010. ID: 1.

[62] VMWare. *Network throughput in a virtual infrastructure*. Whitepaper, 2005.

[63] W3C. "Extensible Markup Language (XML)", 2012. URL http://www.w3.org/XML/.

[64] White, F. E. "Data fusion lexicon". Joint Directors of Laboratories, Technical Panel for C3, Data Fusion Sub-Panel, 1991.

[65] Wu, Z., D. Xiao, H. Xu, X. Peng, and X. Zhuang. "Automated Intrusion Response Decision Based on the Analytic Hierarchy Process". *Knowledge Acquisition and Modeling Workshop, 2008. KAM Workshop 2008. IEEE International Symposium on*, 574–577. 2008. ID: 1.

[66] Yang, S. J., J. Holsopple, and D. Liu. "Elements of impact assessment: a case study with cyber attacks". 73520D–73520D–8, 2009. URL +http://dx.doi.org/10.1117/12.818395.

[67] Ying, L., L. Bingyang, and W. Huiqiang. "Dynamic awareness of network security situation based on stochastic game theory". *Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on*, 101–105. 2010. ID: 1.

[68] Zan, X., F. Gao, J. Han, X. Liu, and J. Zhou. "NAIR: A novel automated intrusion response system based on decision making approach". *Information and Automation (ICIA), 2010 IEEE International Conference on*, 543–548. 2010. ID: 1.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE (DD–MM–YYYY)	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21-03-2013	Master's Thesis	Sep 2011 – Mar 2013

4. TITLE AND SUBTITLE

Modeling Cyber Situational Awareness through Data Fusion

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Raulerson, Evan L., Captain, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/ENY)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-13-M-41

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Dr. Richard Fedors
AETC AFRL/RISF
525 Brooks Road
Rome, NY 13441-4505
(315) 330-3608
richard.fedors@rl.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

Cyber attacks are compromising networks faster than administrators can respond. Network defenders are unable to become oriented with these attacks, determine the potential impacts, and assess the damages in a timely manner. Since the observations of network sensors are normally disjointed, analysis of the data is overwhelming and time is not spent efficiently. Automation in defending cyber networks requires a level of reasoning for adequate response. Current automated systems are mostly limited to scripted responses. Better defense tools are required. This research develops a framework that aggregates data from heterogeneous network sensors. The collected data is correlated into a single model that is easily interpreted by decision-making entities. This research proposes and tests an impact rating system that estimates the feasibility of an attack and its potential level of impact against the targeted network host as well the other hosts that reside on the network. The impact assessments would allow decision makers to prioritize attacks in real-time and attempt to mitigate the attacks in order of their estimated impact to the network. The ultimate goal of this system is to provide computer network defense tools the situational awareness required to make the right decisions to mitigate cyber attacks in real-time.

15. SUBJECT TERMS
computer network, cyber, data fusion, situational awareness, model

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Kenneth Hopkinson
U	U	U	UU	130	19b. TELEPHONE NUMBER (Include Area Code) (937)255-3636, ext 4579

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18